

**MATHEMATICAL AND DATA-DRIVEN PATTERN REPRESENTATION WITH
APPLICATIONS IN IMAGE PROCESSING, COMPUTER GRAPHICS, AND
INFINITE DIMENSIONAL DYNAMICAL DATA MINING**

A Dissertation
Presented to
The Academic Faculty

By

Yuchen He

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Mathematics
College of Sciences

Georgia Institute of Technology

May 2021

© Yuchen He 2021

**MATHEMATICAL AND DATA-DRIVEN PATTERN REPRESENTATION WITH
APPLICATIONS IN IMAGE PROCESSING, COMPUTER GRAPHICS, AND
INFINITE DIMENSIONAL DYNAMICAL DATA MINING**

Thesis committee:

Dr. Sung Ha Kang
School of Mathematics
Georgia Institute of Technology

Dr. Jean-Michel Morel
Centre Borelli, Univ. Paris-Saclay
École Normale Supérieure Paris-Saclay

Dr. Wenjing Liao
School of Mathematics
Georgia Institute of Technology

Dr. Haomin Zhou
School of Mathematics
Georgia Institute of Technology

Dr. Yingjie Liu
School of Mathematics
Georgia Institute of Technology

Date approved: April 9, 2021

A mathematician, like a painter or poet, is a maker of patterns. If his patterns are more permanent than theirs, it is because they are made with ideas.

G.G. Hardy

For my father Wenhao He and my mother Chunlian Ren

ACKNOWLEDGMENTS

I would like to thank the members of my thesis committee for their help in preparation of this work. Specifically, I am grateful for having Professor Sung Ha Kang as my Ph.D. advisor, who throughout the past 4 years have helped me to grow as a young researcher. Her contagious passion for mathematical research and positive personality cast considerable influence on me. Without her patience and open-mindedness, I would have never developed such a broad experience with diverse applications, and this thesis would have never been this rich. Moreover, I would like to show my sincere appreciation for the generous helps from Professor Jean-Michel Morel. During the difficult time of pandemic, he supported my research and offered me precious opportunities. As a well-established researcher, his academic advice has inspired me tremendously.

Special thanks are due to my friends, and colleagues: Dr. Hao Liu, Dr. Martin Húska, Jaemin Park, Christina Giannitsi, and many others. Finally, my research project on deep learning is not possible without the valuable resources supported by Professor Xiaoqun Zhang from Shanghai Jiaotong University.

The author gratefully acknowledges the support by Chateaubriand Fellowship, Embassy of France in United States and Larry O'Hara Fellowship, College of Science, Georgia Institute of Technology. Any views and conclusions contained herein are those of the author, and do not necessarily represent the official positions, express or implied, of the funders.

TABLE OF CONTENTS

Acknowledgments	v
List of Tables	xiv
List of Figures	xvii
Summary	1
Chapter 1: Introduction	2
1.1 What is Pattern?	2
1.1.1 Diverse Forms of Pattern	2
1.1.2 Pattern Formation, Recognition, and Representation	4
1.2 Mathematical Pattern Representation	6
1.2.1 Model-based Approach	7
1.2.2 Data-driven Approach	9
1.3 Organization of the Thesis	11
1.3.1 Contents of Part I (Chapter 2-5)	11
1.3.2 Contents of Part II (Chapter 6-7)	14
1.3.3 Reading Suggestions	15
Chapter 2: Symmetries and Metric Structures in Lattice Patterns	17

2.1	Preliminaries and Notations	18
2.2	Lattice Feature Descriptors β and ρ	22
2.2.1	Equivalence Classes of Shape Descriptor ρ	25
2.2.2	Equivalence Conditions for Scale Descriptors	27
2.3	From Descriptors to Lattice Metric Space $(\mathcal{L}, d_{\mathcal{L}})$	29
2.3.1	Definition of Lattice Metric Space	29
2.3.2	Sub-lattices and Parent-lattices in the Lattice Space	34
2.4	Validation of the Lattice Space \mathcal{L} and Metric $d_{\mathcal{L}}$	38
2.4.1	Visual Validation	38
2.4.2	Quantitative Validation	39
2.5	Application to Error Quantification of Lattice Identification and Separation Algorithm (LISA)	41
2.5.1	Variational Model for Lattice Separation	43
2.5.2	Lattice Identification and Separation Algorithm (LISA)	44
2.5.3	Analytical Properties of LISA: Superlattice and Spectrum Surface	48
2.5.4	Robustness of LISA against Gaussian Perturbation	51
2.5.5	Numerical Experiments with Various Superlattice Patterns	53
2.6	Application to Grain Defect Detection	62
2.6.1	Lattice Clustering Algorithm based on Lattice Metric Space	64
2.6.2	Numerical Experiments on Grain Defect Detection	65
2.7	Summary	72
Chapter 3: PDE-based Shape Representation and Vectorization		74
3.1	Region-based Representation – Shape Skeleton	75

3.2	Hamilton-Jacobi Skeleton Algorithm	76
3.2.1	Distance Transform using the Fast Sweeping Algorithm	76
3.2.2	Computation of Average Outward Flux	80
3.2.3	Point Classification based on Local Topology	82
3.2.4	Homotopy Preserving Thinning	84
3.3	Numerical Experiments on Shape Skeletons	85
3.3.1	Skeletonization of 2D Shapes	87
3.3.2	Effects of the Parameter γ	87
3.3.3	Shape Reconstruction from Medial Axis	92
3.3.4	Performance of Distance Computation	95
3.4	Contour-based Representation – Silhouette Vectorization	99
3.5	Outline of the Affine-scale Space Vectorization Procedure	101
3.6	Sub-pixel Curvature Extrema Localization	103
3.7	Affine Scale-space Control Points Identification	104
3.7.1	Backward Tracing via Inverse Affine Shortening Flow	104
3.7.2	Degenerate Case	107
3.8	Adaptive Cubic Bézier Polygon Approximation	108
3.8.1	Bézier Fitting with Chord-length Parametrization	108
3.8.2	Control Point Refinement: Deletion of Sub-pixel Extrema	109
3.8.3	Control Point Refinement: Insertion for Accuracy	110
3.9	Numerical Experiments on Silhouette Vectorization	111
3.9.1	Data Preparation and Parameter Settings	111
3.9.2	General Performance	112

3.9.3	Tests on Degenerate Cases	114
3.9.4	Effect of the Error Threshold τ_e	116
3.9.5	Effect of the Smoothness Parameter σ_0	117
3.9.6	Qualitative Comparison with Feature Point Detectors	117
3.9.7	Quantitative Comparison with Feature Point Detectors	120
3.9.8	Comparison with State-of-the-art Software	121
3.9.9	Quantitative Study of Efficiency and Accuracy	122
3.10	Summary	123
Chapter 4: Submanifold Representation Induced by Point Cloud		127
4.1	Surface Identification via Minimizing Distance-weighted Surface Area . . .	129
4.1.1	Energy by Distance-weighted Surface Area	129
4.1.2	Semi-Implicit Method (SIM)	130
4.1.3	Augmented Lagrangian Method (ALM)	133
4.1.4	Connection between SIM and ALM Algorithms	136
4.1.5	Implementation Details	137
4.2	Numerical Experiments on Model of Distance-weighted Surface Area . . .	139
4.2.1	General Performance on 2D and 3D Point Clouds	139
4.2.2	Choice of Parameters for ALM and the Effects	144
4.3	Curvature-regularized Energy and Its Fast Optimizing Algorithms	149
4.3.1	Curvature Regularized Surface Reconstruction Model	150
4.3.2	Analytical Aspects	152
4.3.3	Operator Splitting Method (OSM)	156

4.3.4	Augmented Lagrangian Method (ALM)	160
4.3.5	Implementation Details	164
4.4	Numerical Results and Comparisons	169
4.4.1	Choice of Parameters for ALM Method	169
4.4.2	Comparison between OSM and ALM	171
4.4.3	Effect of curvature constraint: OSM with $s = 2$	173
4.4.4	Three Dimensional Examples	176
4.5	Summary	178
Chapter 5: Complementary Adaptation in Underwater Color Correction		183
5.1	CIELAB Color Space	186
5.1.1	Basic Notions of CIELAB	186
5.1.2	CIELAB Boundary Estimation	188
5.2	Complementary Adaptation Model in CIELAB	189
5.2.1	Tikhonov-type Optimization in CIELAB	189
5.2.2	Robust Hue-preserving Image Enhancement	193
5.2.3	Improvement on the Uniformity of CIELAB	195
5.3	Numerical Experiments	197
5.3.1	General Examples	197
5.3.2	Different Underwater Color Cast	198
5.3.3	Necessity of the Robust Factor	200
5.3.4	Behaviors of the Saturation Parameter η	201
5.3.5	Qualitative Comparison	203

5.3.6	Quantitative Evaluation and Comparison	204
5.4	Conclusion	207
Chapter 6: Automatic PDE Identification from Noisy Data		208
6.1	Data Organization and Denoising	210
6.1.1	Data Organization and Notations	210
6.1.2	Noise Amplification during Differentiation	212
6.1.3	Successively Denoised Differentiation (SDD)	213
6.2	PDE Model Identification Methods: ST and SC	215
6.2.1	Subspace Pursuit Time Evolution (ST)	216
6.2.2	Subspace Pursuit Cross Validation (SC)	222
6.3	Numerical Experiments on Robust PDE Identification	225
6.3.1	Transport Equation	227
6.3.2	Burgers' Equation	231
6.3.3	Burgers' Equation with Diffusion	233
6.3.4	The KdV Equation	235
6.3.5	A Larger Dictionary	236
6.3.6	Two Dimensional PDEs	237
6.3.7	Identifiability Based on the Given Data	238
6.3.8	Choice of Smoother in SDD	240
6.4	Support Recovery in Statistics	241
6.5	PDE Identification via ℓ_1 -PsLS	243
6.5.1	Problem Setting	243

6.5.2	Local-Polynomial Regression Estimators for Derivatives	244
6.5.3	ℓ_1 -regularized Pseudo Least Square Model	245
6.6	Recovery Theory for ℓ_1 -PsLS based PDE Identification	247
6.6.1	Signed-Support Recovery	247
6.6.2	Assumptions	248
6.6.3	Statement of Main Result	249
6.6.4	Proof Strategy of the Main Theorem	251
6.7	Analysis Under Sample Incoherence Matrix Assumptions	252
6.7.1	Statement of Proposition	252
6.7.2	Proof Overview of Proposition 6.7.1	253
6.7.3	Technical Challenges	255
6.8	Uniform Convergence of Sample Incoherence Matrix	256
6.9	Numerical Experiments	258
6.9.1	Experimental Setting	258
6.9.2	Numerical Verifications of Main Statements	260
6.9.3	Impact of β_{\min}^* in Signed-Support Recovery of ℓ_1 -PsLS	261
6.10	Summary	262
Chapter 7: Deep Spatial-temporal Synthesizer for dynamic PET Reconstruction		265
7.1	Workflow Overview	266
7.2	Nonnegative Matrix Factorization for dPET	267
7.2.1	Imaging Model for Sparsely Sampled dPET	267
7.2.2	NMF Reconstruction	268

7.3	Proposed Model	269
7.4	Numerical Experiments	272
7.4.1	Interpretability of Low-rank Bases	274
7.4.2	Performance on TAC Reconstruction	274
7.4.3	Tests on Hyperparameters	275
7.4.4	Qualitative Comparison	276
7.4.5	Quantitative Comparison	280
7.5	Summary	282
Chapter 8:	Conclusion	283
Appendices	286
Appendix A:	Appendix for Chapter 2	287
Appendix B:	Appendix for Chapter 3	288
Appendix C:	Appendix for Chapter 6	290
References	318
Vita	355

LIST OF TABLES

2.1	Lattice Clustering Algorithm	64
3.1	Comparative measures of the reconstruction results in Figure 3.7 ($\varepsilon = 1.5$). Higher values of J and DSC indicate higher similarity between S_o and S_r . When Bpn is positive (negative), S_r is an over- (respectively under-) coverage for S_o , and when $Bpn = 0$, there is no bias. We report these measures when $\varepsilon = 0$ for comparison.	95
3.2	Performance of the proposed method applied to examples in Figure 3.11. The compression ratios are displayed for PNG and JPG, respectively. *We note that the PNG image in (a) has a single channel, thus converting it to JPG increases the size.	113
3.3	Comparison with image vectorization software in terms of the number of control points. We compared with Vector Magic (VM), Inkspace (IS), and Adobe Illustrator 2020 (AI). For VM, we report the number of control points using three settings: High/Medium/Low. For AI, the values with dagger [†] indicate the numbers of control points produced by the automatic simplification. The input image dimensions are 581×564 , 625×598 , 400×390 , 903×499 , 515×529 , and 1356×716 from top to bottom. We also report the mean relative reduction (MRR) of the number of control points computed for the results above.	124
4.1	CPU time (s) for SIM, ALM using $r = 0.5, 1, 1.5$, and 2, and the explicit method in [285] with $\Delta t = 20$ for the point cloud data sets in Figure 4.3. Both SIM and ALM shows fast convergence.	142
4.2	CPU time (s) of SIM and ALM compared to the explicit method in [285] for the point cloud data sets of Figure 4.4. Both SIM and ALM show fast convergence.	142

6.1	The procedure of SDD, where the spatial and time smoothing operators $S_{(\mathbf{x})}$ and $S_{(t)}$ are defined in Equation 6.7 and Equation 6.8 respectively. The operator D_t given in Equation 6.5 represents numerical time differentiation by the forward difference scheme, and $D_{x_i}U$ for $i = 0, 1, \dots, N - 1$ represents numerical spatial differentiation with respect to x_i given by the 5-point ENO scheme [387].	215
6.2	Identification of the transport equation (Equation 6.15) with different noise levels. In the noise-free case, applying SDD does not introduce a strong bias. The identification results (second column) by ST and SC are stable even with 30% noise. Here $w = 20$ for ST, and $\alpha = 1/200$ for SC.	228
6.3	Identification of the transport equation (Equation 6.15) with the discontinuous initial condition (Equation 6.16) and different noise levels. In the noise-free case, applying SDD does not introduce strong bias. The identification results (second column) by ST and SC are stable even with 30% noise. Here $w = 20$ for ST, and $\alpha = 1/200$ for SC.	231
6.4	Identification of the Burgers' equation (Equation 6.17) with initial condition (Equation 6.18) and different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 40%. Here $w = 20$ for ST, and $\alpha = 1/500$ for SC.	232
6.5	Comparison of ST, SC with IDENT in [369] and the method in [367] for the identification of the Burgers' equation (Equation 6.17) with the initial condition (Equation 6.19), and various noise levels. In this table, we only include the reconstructed terms with the coefficient magnitudes above 10^{-2} . ST, SC and IDENT are very stable compared to the method in [367]. The coefficient error e_c (Equation 6.13) and the time evolution error e_e (Equation 6.14) are shown. The errors given by ST, SC and IDENT are smaller than the errors given by the method in [367]. Comparison of ST, SC with IDENT in [369] and the method in [367] for the identification of the Burgers' equation (Equation 6.17) with the initial condition (Equation 6.19), and various noise levels. This table only includes the reconstructed terms with the coefficient magnitudes above 10^{-2} . ST, SC, and IDENT are very stable compared to the method in [367]. The coefficient error e_c (Equation 6.13) and the time evolution error e_e (Equation 6.14) are shown. The errors given by ST, SC, and IDENT are smaller than the method's errors in [367].	234
6.6	Identification of the Burgers' equation with diffusion (Equation 6.20) with different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 5%. Here $w = 20$ for ST, and $\alpha = 1/10$ for SC.	234

6.7	Identification of the KdV equation (Equation 6.21). Both ST and SC can identify the correct PDE.	236
6.8	Identification of Equation 6.22 with different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 5%. Here $w = 20$ for ST, and $\alpha = 1/500$ for SC. . .	237
6.9	Identification of the two dimensional PDE (Equation 6.24) with different noise levels. The identification results (second column) by ST and SC have small e_c and e_r for a noise level up to 10%. Here $w = 10$ for ST, and $\alpha = 3/200$ for SC.	238
6.10	Specific choices of the constants in the order of $h_N = \Theta(N^{-\frac{1}{7}})$ and $w_M = \Theta(M^{-\frac{1}{7}})$ for the experiments on Viscous Burgers equation and KdV equation are presented.	260
7.1	Comparison among proposed model with various combinations of hyper-parameters: number of bases (K) and number of iterations (I).	276
7.2	Quantitative comparison (Mean \pm Std.) of different methods' performances on the testing dataset (39 samples). For the results of quality evaluation, the best ones are bolded, and the second best* ones are marked with asterisks. Deep learning based methods were trained using the same training dataset (153 samples) on a common machine configuration.	281
7.3	Training and testing efficiency.	281
B.1	Silhouette dataset used in the experiments. The last four are used in Figure Figure 3.13 for computing the average $\rho(\tau_e)$. These silhouettes are chosen from [253], which are released under Creative Commons CC0. . . .	289

LIST OF FIGURES

- 2.1 Equivalent bases and the minimal basis. (a) $\Lambda(3, 4i)$, (b) $\Lambda(4i, -3+4i)$, and (c) $\Lambda(-3-4i, -6-4i)$ represent an identical lattice. (a) $(3, 4i)$ is a minimal basis: $|\operatorname{Re}(\frac{4i}{3})| = 0 < \frac{1}{2}$. (b) $(4i, -3+4i)$ is not minimal: $|\operatorname{Re}(\frac{-3+4i}{4i})| = 1 > \frac{1}{2}$, and (c) $(-3-4i, -6-4i)$ is not positive: $\operatorname{Im}(\frac{-6-4i}{-3-4i}) = -\frac{12}{25} < 0$ 20
- 2.2 Effects of changing β and ρ . (a) $\Lambda\langle 1, i \rangle$, (b) $\Lambda\langle 2, i \rangle$, (c) $\Lambda\langle e^{i\pi/6}, i \rangle$, (d) $\Lambda\langle 1, 2i \rangle$, (e) $\Lambda\langle 1, e^{2\pi i/3} \rangle$, and (f) $\Lambda\langle 2, e^{2\pi i/3} \rangle$. From (a) to (b), β is changed from 1 to 2. From (a) to (c), β is rotated. From (a) to (d), ρ is changed from i to $2i$. From (a) to (e), ρ is rotated. From (a) to (f), both β and ρ are changed. 23
- 2.3 Region \mathcal{P} and the fundamental set of Γ -actions. (a) \mathcal{P} is the gray region including the boundary represented in \mathbb{C} . ρ , ρ' and ρ'' are the shape descriptors for $\Lambda(3, 4i)$, $\Lambda(4i, -3+4i)$, and $\Lambda(-3-4i, -6-4i)$ respectively from in Figure 2.1. Since $4i/3 \in \mathcal{P}$, $(3, 4i)$ is a positive minimal basis. (b) A fundamental set of the modular group Γ acting on the upper half plane. If $\operatorname{Re}(\rho) = -1/2$, ρ and $\rho + 1$ are in the same orbit of a Γ -action. 25
- 2.4 Examples of subspaces of \mathcal{L} . For any $\beta \in \mathcal{K}$, (a) shows a square lattice $\Lambda\langle \beta, i \rangle$. The red and blue arrows indicate two directions. Stretching $\Lambda\langle \beta, i \rangle$ along them represents two different families of lattices. They form a subspace of \mathcal{L} shown in (b), which is homeomorphic to \mathbb{R} as in (c). (d) shows a hexagonal lattice $\Lambda\langle \beta, e^{i\pi/3} \rangle$. Stretching it along the three marked directions generates three distinct families of lattices. (e) is the subspace they form in \mathcal{L} , which is homeomorphic to the structure in (f). 30
- 2.5 An illustration of the 8 types of paths, D_1 – D_8 in Equation 2.10 connecting (β, ρ) and (β', ρ') via 4 extra points in $\{(\beta_0, \rho_0) \mid \beta_0 \in \mathcal{K}, |\rho_0| = 1, \rho_0 \in \mathcal{P}\}$. 33
- 2.6 The lattice space \mathcal{L} is a product space $\mathcal{K}/\sim_1 \times \mathcal{P}/\sim_2$ modulo \sim_3 . The distance $d_{\mathcal{L}}((\beta, \rho), (\beta', \rho'))$ is the minimal length of the paths connecting (β, ρ) and (β', ρ') when the distance between equivalent points is reduced to 0. Here the green line shows D in Equation 2.8, the red line is D_3 in Equation 2.10, and the blue line is D_4 in Equation 2.10. Since D satisfies the triangle inequality, D is shorter than D_4 33

- 2.7 One-to-one correspondence between the sub- and the parent-lattices. (a) A lattice $\Lambda\langle\beta, \rho\rangle = \Lambda\langle 14.7721 + 2.6047i, e^{i\pi/3}\rangle$. (b) A sub-lattice $\Lambda\langle\beta, 2\rho+1\rangle$ in white, with lattice (a) in gray. (c) A parent-lattice $\Lambda\langle\beta/2, 2\rho+1\rangle$ of (a). The common particles are emphasized with white color. 37
- 2.8 Metric comparison. Lattice (a) $\Lambda_A = \Lambda(11.8177 + 2.0838i, -2.1706 + 12.3101i)$ and (b) $\Lambda_B = \Lambda(2.0838 - 11.8177i, 12.3101 + 2.1706i)$ are visually similar. The 4-tuple measure indicates a significant difference in θ , while $d_{\mathcal{L}}$ gives a small value. The lattices (a), (c) $\Lambda_C = \Lambda(-1.1766 + 13.4486i, -2.0838 + 11.8177i)$ and (d) $\Lambda_D = \Lambda(11.8177 + 2.0838i, -2.1706 + 12.3101i)$ are more distinguishable, but the differences are scattered in four numbers using (Equation 2.16). $d_{\mathcal{L}}$ integrates these differences and provides a compact measure. 39
- 2.9 Visual effects of $d_{\mathcal{L}}$. Five different lattices: (a) $\Lambda_A = \Lambda\langle 11, e^{i\pi/3}\rangle$, (b) $\Lambda_B = \Lambda\langle 11, e^{i\pi/2}\rangle$, (c) $\Lambda_C = \Lambda\langle 13, e^{i\pi/2}\rangle$, (d) $\Lambda_D = \Lambda\langle 11, e^{i61\pi/180}\rangle$, and (e) $\Lambda_E = \Lambda\langle 13, e^{i61\pi/180}\rangle$ are displayed. Pairwise distances: $d_{\mathcal{L}}(\Lambda_A, \Lambda_B) = 0.5493$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_C) = 0.7083$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_D) = 0.0203$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_E) = 0.4477$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_C) = 0.4472$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_D) = 0.5293$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_E) = 0.6929$, $d_{\mathcal{L}}(\Lambda_C, \Lambda_D) = 0.6929$, $d_{\mathcal{L}}(\Lambda_C, \Lambda_E) = 0.5293$, and $d_{\mathcal{L}}(\Lambda_D, \Lambda_E) = 0.4472$ are computed. They are consistent with the visual perception of the lattice differences. 40
- 2.10 Properties of $d_{\mathcal{L}}$. (a) Effect of changing w . (b) Misorientation of hexagonal lattices is emphasized using $d_{\mathcal{L}}$, which corresponds to the left and right edges. (c) High symmetry of the hexagonal lattice is reflected by the symmetry of the blue curve. Lattices to be compared are not necessarily of the same type, and $d_{\mathcal{L}}$ considers the lattice equivalence relations. 41
- 2.11 Challenges of pattern separation. Each image above has two lattices superposed. (a) The red boxes indicate textons of a single lattice, and they have different interiors which can confuse the texton-based methods. (b) Using non-superposed lattice identification methods, wrong local features (e.g., L-shapes [115], shown as the red arrows) can be identified. These red arrows do not correspond to any of the true underlying lattices. (c) The pink and the yellow L-shapes in the upper-left corner denote the true lattice components. The moiré patterns indicated by the red, blue, and green regions are different from the underlying lattices. 42

- 2.12 Steps of LISA. (A) An image processed by (Equation 2.20). **Step 1:** (B) The power spectrum on the polar coordinate, and the high responses using $J = 5$. (C) Peak locations refined via matching Gaussian impulses. **Step 2:** (D) Generate lattice candidates $\mathcal{T}_{\mu_{k,l}}\Lambda_{(k,l)}$, $k, l = 1, \dots, 5, k \neq l$, for each pair of high peaks, and compute their energies (Equation 2.21). Pick (x_3, x_5) (red and purple in (B)) to be the optimal $\mathcal{T}_{\mu_1}\Lambda_1$, since it has the lowest energy. **Step 3:** (Optional) (E) Update $\mathcal{T}_{\mu_1}\Lambda_1$ with $\mathcal{T}_1\Lambda_{\mu_1}^{(5)}$. **Step 4:** (F) The optimal lattice $\mathcal{T}_{\mu_1}\Lambda_1$ identified in this iteration; and the absolute difference between $\mathcal{T}_{\mu_1}\Lambda_1$ and the underlying true lattice. The absolute difference has an average value of 0.0202, and maximum of 0.1924, showing the effectiveness of LISA. (G) The remainder image. The average intensity 0.0710 is greater than the accuracy criterion 0.01; thus, proceed to the next iteration. 45
- 2.13 Effect of relative translation. (a) A superlattice composed of $\mathcal{T}_{4-3i}\Lambda\langle 12, i \rangle$ and $\mathcal{T}_{-4+3i}\Lambda\langle 12, i \rangle$. (b) The power spectrum of (a) where peaks are missing due to the relative translations. From this incomplete reciprocal lattice, LISA identifies lattice (c) and (d) each shown in white, superposed over (a) in gray. 50
- 2.14 LISA's robustness against Gaussian perturbation. In the first column, a single lattice $\mathcal{T}_0\Lambda\langle 12, e^{i\pi/18} \rangle$ is shown in (a) with its power spectrum surface in (d). A centered Gaussian perturbation is applied with standard deviation (b) $s = 0.5$ and (c) $s = 1$, and their power spectra are displayed in (e) and (f), respectively. Notice that in the frequency domain, the reciprocal bases away from the origin are smeared by noises, but those near the origin remain high responses. The lattices identified by LISA in (b) and (c) are robust against the perturbation; their distances to (a) are 0.0046 and 0.0081, respectively. 52
- 2.15 A typical example of LISA. (a) A superlattice of three lattices: $\mathcal{T}_{2-4i}\Lambda\langle -9.9927 + 0.0315i, 1.0014e^{i17\pi/36} \rangle$, $\mathcal{T}_{-7-4i}\Lambda\langle -4.4820 + 12.1815i, i \rangle$ and $\mathcal{T}_{1-5i}\Lambda\langle -4.9898 - 8.5389i, 1.0298e^{i7\pi/12} \rangle$. (b)–(d) display the lattices identified by LISA. Each metric value shows the distance between the true lattice and the identified one in \mathcal{L} 53
- 2.16 Superlattice with more layers. (a) A superlattice of 5 lattices: $\mathcal{T}_{2-5i}\Lambda\langle 11, e^{i7\pi/18} \rangle$, $\mathcal{T}_{3+4i}\Lambda\langle 11.7378 + 2.4949i, i \rangle$, $\mathcal{T}_0\Lambda\langle 3.7082 + 11.4127i, e^{4\pi/9} \rangle$, $\mathcal{T}_{1-2i}\Lambda\langle 14.0954 + 5.1303i, i \rangle$, and $\mathcal{T}_0\langle 11.8177 + 2.0838i, i \rangle$. (b)–(f) show the extracted patterns using LISA. Notice that all the metric values $d_{\mathcal{L}}(\hat{\Lambda}, \Lambda)$ comparing the true lattices with the identified ones are very small. 54

- 2.17 Mixture of translational lattices. (a) A superlattice of four lattices: $\mathcal{T}_0\Lambda\langle 12, i \rangle$, $\mathcal{T}_{1+i}\Lambda\langle 11.8177+2.0838i, i \rangle$, $\mathcal{T}_{2-3i}\Lambda\langle 12, i \rangle$, and $\mathcal{T}_{2-5i}\Lambda\langle 11.8177+2.0838i, i \rangle$. (b)–(e) show the identified patterns by LISA. 55
- 2.18 Close particles. (a) A superlattice of three lattices obtained by translating $\mathcal{T}_0\Lambda\langle 14.7721 + 2.6047i, i \rangle$ by $4 - 2i$, $1 - 2i$ and $2 - 5i$. These translations push particles close, and generate a pattern whose lattice points are composed of three dots. (b)–(d) show that LISA successfully distinguishes them with high precision as indicated by the values of $d_{\mathcal{L}}$ 56
- 2.19 Incomplete lattice. (a) A superlattice composed of a complete lattice $\mathcal{T}_0\Lambda\langle 11.6924 + 2.6994i, e^{i4\pi/9} \rangle$, shown in (b), and a portion of $\mathcal{T}_{2-3i}\Lambda\langle 11.8177+2.0838i, i \rangle$, shown in (c). (d) and (e) are the identified patterns by LISA (in white) over the original (a) (in gray). (f) $\min(\mathcal{T}\hat{\Lambda}_2, I)$, where $\mathcal{T}\hat{\Lambda}_2$ is the identified lattice in (e) and I is the original image in (a). This shows the intersection of (a) and (e). 57
- 2.20 Importance of the density restriction. (a) A superlattice of $\mathcal{T}_{2-10i}\Lambda\langle 10, e^{i17\pi/36} \rangle$ and $\mathcal{T}_{-3+5i}\Lambda\langle 9.9756+0.6976i, e^{i17\pi/36} \rangle$. Without the second term in (Equation 2.21), we obtain a dense lattice $\mathcal{T}\hat{\Lambda}$ in (b). With the density restriction, we get $\mathcal{T}\tilde{\Lambda}$ in (c) which is the correct lattice pattern. (d) compares (b) and (c), where the white pixels are $\mathcal{T}\tilde{\Lambda} \cap \mathcal{T}\hat{\Lambda}$ (the particles commonly captured by (b) and (c)), the green are $\mathcal{T}\tilde{\Lambda} - \mathcal{T}\hat{\Lambda}$ (the extra points in (b) compared to (c)), and the red are $\mathcal{T}\hat{\Lambda} - \mathcal{T}\tilde{\Lambda}$ (particles in (c) not covered by (b)). It shows that (c) is almost a sub-lattice of (b). (e) $\min\{\mathcal{T}\tilde{\Lambda}, I\}$ showing the intersection of (a) and (b). This shows that the dense lattice (b) approximates the moiré pattern at the center of (a) 58
- 2.21 Flake-like pattern generated by lattices. (a) A flake-like superlattice of hexagonal lattices with β equal to 10, 13, 15 and 12. In the same order, (b)–(e) show LISA successfully identifies the underlying lattices. 59
- 2.22 Flower pattern generated by lattices. (a) A flower superlattice of four lattices with scale descriptors having a common norm $|\beta| = 11$, and inclination angles equal to 53° , 143° , -53° and -143° . (b)–(e) show the lattices identified with high precision by LISA. 60
- 2.23 LISA on real images. (a) and (c) are the images of TMD monolayers adjusted from [163] 3 and [164] 1 (c), respectively. (b) and (d) show the identified lattice patterns, and lattice points from different layers are colored in red and green, respectively. 61

2.24	LISA on grain segmentation. (a) A grain image adjusted from [166] 15 (a). (b) $\mathcal{T}_{-1.3794+9.7510i}\Lambda\langle -10.9881 - 12.1163i, -0.4579 + 0.8950i \rangle$ and (c) $\mathcal{T}_{9.6287+9.5640i}\Lambda\langle -15.7326 - 4.7420i, 0.4813 + 0.8800i \rangle$ are the lattice patterns identified by LISA. (d) Particles shared in (a) and (b) are colored in green, and those shared with (c) in red. The white particles are shared by the lattices in (b) and (c).	61
2.25	CPU time of LISA. Fixing two lattices, the base image width is $m = 119$, $K = 10$, and $J = 6$. (a) The image width m is increasing while K and J are fixed. (b) The number of iteration K is increasing with m and J fixed. (c) The number of connected components J is increasing while keeping m and K fixed. Roughly, LISA depends linearly on J and K respectively, and quadratically on m	62
2.26	Direct classification using $d_{\mathcal{L}}$. (a) PFC image from [104] Fig. 4. (b) Lattice labels obtained in Step 1 of LCA. (c) zoomed-in partial region from (b). (d) and (f) show Euclidean distance function of (b_1, b_2) representation with respect to that of the red points. (e) and (g) show the lattice distance function $d_{\mathcal{L}}$ of (β, ρ) representations with respect to that of the red points, which are the same as those in (d) and (f) respectively. Linear interpolation is applied to fill the color in (d)–(g).	67
2.27	Apply LCA to the image Figure 2.26(a). Here (a) shows the curve $g(t)$. Results when (b) $T = 0.4$, (c) $T = 0.5$ and (d) $T = 0.8$ show the effect of T	68
2.28	Instability of k-means. (a) Box-plot of the number of grains against parameter K in k-means. (b)–(d) use k-means with different initializations. (b) and (c) set $K = 30$ and $T = 0.5$ and (d) uses $K = 50$ and $T = 0.5$	69
2.29	(a) There are 3 grains: one on the top, one in the middle, and one in the bottom. Image from [102] Fig. 1. (b) The curve $g(t)$ with 3 major jump-discontinuities. (c) $T = 0.5$. (d) $T = 0.8$	69
2.30	(a) There are 2 grains with a regular boundary. Image adapted from [180] Fig. 1(a). (b) The curve $g(t)$ with 2 major jump-discontinuities. (c) Result with $T = 0.4$	70
2.31	(a) There are 2 grains presented and the grain boundary is irregular. Image adapted from [178] Fig. 1. (b) The curve $g(t)$ with 2 major jump-discontinuities and the jump is rough. (c) Result with $T = 0.8$	70
2.32	(a) Grain boundary between non-hexagonal grains. Image adapted from [179] Fig. 3. (b) The curve $g(t)$. (c) Result with $T = 0.7$	71

3.1	Skeletons (black curves) of some elementary shapes.	76
3.2	(a) The normal vectors at the neighboring points used for approximating the flux Equation 3.8 at the central pixel. (b) An example graph G constructed for the pixel P . For any arbitrary pixel, its 8 neighborhoods are indexed as shown here. Neighboring pixels inside the shape (black circles) are the vertices of G , and two vertices are connected if they are 8-neighborhood to each other. We avoid the 3-loops at the corners, e.g., $0 - 1 - 7$, by directly connecting the furthest two among them.	83
3.3	Skeletons (red curves) for various shapes computed by HJS. In all examples above, we used the default parameter $\gamma = 2.5$	88
3.4	Multi-scale representation of the shape using skeletons computed by different γ . (a) $\gamma = 2.5$. (b) $\gamma = 1.5$. (c) $\gamma = 1.2$. By choosing a smaller γ , the identified skeleton becomes more robust against boundary perturbation and captures the large-scale shape features.	90
3.5	HJS with $\gamma < 1$ used as a homotopy classifier. (a) The skeleton is a single point, hence the shape is simply-connected. (b) The skeleton is homeomorphic to a circle, hence the shape is not simply-connected and has genus 1. (c) The skeleton consists of 10 points, hence the shape has ten simply-connected components. In (a) and (c), the identified skeleton points are emphasized by red disks for visualization.	91
3.6	HJS with $\gamma < 1$ used as a deficiency detector in binary shapes. (a) The given shape and identified non-trivial skeleton using $\gamma = 0.9$. (b) A hole on the boundary of the top-left petal. (c) A hole on the bottom-right pedal. (b) and (c) show the deficiencies inducing the non-trivial skeleton in (a). In all examples here, we keep $\gamma = 0.9$	91
3.7	Shape reconstructed from the medial axis transform. (a) Original shapes. (b)-(d) The shapes reconstructed from the HJS using (b) $\gamma = 0.9$, (c) $\gamma = 2.5$, and (d) $\gamma = 20$. Here we fixed $\varepsilon = 1.5$	93
3.8	Effects of varying ε on the comparative measures. Here we plot the values of the rescaled measures, \bar{J} , \overline{DSC} and $ \overline{Bpn} $, against different values of ε , when HJS ($\gamma = 2.5$) is applied to the <i>sakura</i> in the first row and the <i>trophy</i> in the third row of Figure 3.7. Both plots indicate that using slightly dilated disks improves the reconstruction results.	96

3.9	(a) Binary image (537×700): a cat silhouette. The distance function is computed by (b) a brute-force method (algorithm 4), (c) the fast sweeping algorithm (algorithm 2), and (d) the F-H algorithm [207]. Brighter pixels indicates further distance from the contour. The F-H algorithm is the fastest, then the fast-sweeping, and the brute-force is the slowest. (e) shows the skeleton computed based on the distance transform in (b); (f) shows the skeleton computed based on the distance transform in (c); and (g) shows the skeleton computed from (d). In all cases, we fixed $\gamma = 2.5$	98
3.10	A flowchart of the proposed method. (a) A given raster image of a <i>cat</i> 's silhouette. (b) Zoom-in of (a). (c) Extracted bilinear outline of (a). (d) Inversely tracing the curvature extrema along the affine shortening flow. (e) The vectorized outline of (a) with control points marked as red dots. (f) Zoom-in of (e). (g) Vectorized result of silhouette (a) by the proposed method. (h) Zoom-in of (g). Notice the improvement from the given raster image (a) to the proposed method's result in (g), as well as the zoom of (b) and (h).	102
3.11	General performance. (a) <i>Cat</i> and (b) its vectorized outline (42 control points). (c) <i>Butterfly</i> and (d) its vectorized outline (158 control points). (e) Text design and its vectorized outline (2683 control points). Each red dot signifies the location of a control point. (g) Two letters exerted from (e) scaled up with the same magnitude. (h) Zoom-in of the vectorization (f) on the two letters in (g).	113
3.12	Degenerate cases. In (a) and (c), no candidate control points were identified. Our algorithm handles such situations by checking if the outline is a circle. If it is, e.g. (a), the center and radius are computed, and a circle is drawn without Bézier fitting; hence, there is no control point (red dots) on the vectorized outline (b). The blue dot indicates the center of the circle. If it is not a circle, e.g., (c), a pair of most distant points are inserted to initiate the Bézier fitting, such as in (d). (e) shows the low-resolution version of (c), and (f) displays its vectorization. When the resolution is low, all the control points are identified curvature extrema. In (g), three of the outline curves are identified as circles, and the others are fitted by Bézier polygons. (h) shows the vectorized result.	115
3.13	(a) For the 20 silhouettes in our data set (Table B.1), the solid curve shows the average relative reduction of the number of control points $\rho(\tau_e)$ Equation 3.20, and the dashed curves indicate the standard deviations. (b) The positive relation between the number of control points when $\tau_e = 10.0$ is large and the number of corners of a silhouette. Each dot represents a sample in our data set. The red curve is computed by linear regression with a goodness of fit $R^2 = 0.75592$	117

3.14	Effect of the smoothing parameter σ_0 . (a) A silhouette of a <i>tree</i> where the boxed region is examined in detail. Vectorization using (b) $\sigma_0 = 2.0$ (362 control points) (c) $\sigma_0 = 1.0$ (448 control points), and (d) $\sigma_0 = 0.5$ (500 control points). With smaller values of σ_0 , the vectorized outline is sharper, and the number of control points increases.	118
3.15	Comparison between the control points (red dots) plus the centers of circles (blue dots) produced by the proposed algorithm and other point feature detectors (green crosses). (a) Compared with the Harris corner detector [255]. (b) Compared with the FAST feature detector [256]. (c) Compared with the SURF detector [257]. (d) Compared with the SIFT detector [258]	119
3.16	Repeatability ratios of the methods in comparison when the silhouettes in the first column are rotated or scaled. Notice that the blue lines (proposed method) are near 1. The performance of our method is the most consistent across these different silhouettes.	121
3.17	Comparison among the given raster image (red boxes), AI (orange boxes), the proposed with $\sigma_0 = 1$, $\tau_e = 1$ (green boxes), and the proposed with $\sigma_0 = 0.1$, $\tau_e = 0.5$ (blue boxes). With smaller numbers of control points ($\#C$), our method preserves better the geometric details of the given silhouette.	124
3.18	(a) Comparison between AI ($\gamma = 150^\circ$) and the proposed method ($\sigma_0 = 1$) when the complexity parameters (μ for AI, τ_e for ours) vary. The circled dot corresponds to our default setting. (b) Comparison between AI with simplification specified by various combinations of μ and γ , and the proposed method using merging with fixed $\sigma_0 = 0.5$ and varying τ_e . In both figures, smaller dots indicate higher levels of complexity for AI (μ) and the proposed method (τ_e), respectively. A dot locating to the right indicates higher accuracy, and a dot in a lower position implies higher efficiency. . . .	125
4.1	Test point clouds. (a) Five-fold circle (200 points). (b) Jar (2100 points). (c) Torus (2000 points).	128
4.2	The CPU-time (s) of ALM until convergence for the five-fold circle point cloud in Figure 4.1 (a). Here $r = \varepsilon = 1$ and η varies from 0.05 to 0.5. The connection between SIM and ALM indicates that large η slows down ALM. In this graph, as η increases, the time required to reach the convergence increases.	137

- 4.3 The test point clouds: triangle with 150 number of points, ellipse with 100 points, square with 80 points, and five-fold-circle with 200 points. (a) The top row, identical initial condition applied to SIM and ALM for different \mathcal{D} . (b) The middle row, the results obtained by SIM. (c) The bottom row, the results obtained by ALM using $r = 1.5$. Both methods give compatible results. 140
- 4.4 The first row shows ALM and SIM applied to the 3D jar point cloud in Figure 4.1 (b). (a) The result of ALM with $r = 1.3, \varepsilon = 0.5, \eta = 0.6$. (b) The result of SIM. The second row shows the methods applied to the 3D torus point cloud in Figure 4.1 (c). (c) The result of ALM with $r = 1.3, \varepsilon = 0.5, \eta = 0.6$. (d) The result of SIM. Both methods are compatible and shows good results. 141
- 4.5 The effect of the distance function for varying-density point clouds: the face with n_1 points, the head with n_2 points, and each ear with n_3 points. (a) the given point cloud is with $(n_1, n_2, n_3) = (20, 10, 20)$, and shows the 0-level-set of ϕ^n at 15th iteration, (b) $(n_1, n_2, n_3) = (50, 10, 20)$, and shows 18th iteration, and (c) $(n_1, n_2, n_3) = (20, 10, 40)$, and shows 20th iteration. These three curves eventually degenerate to a point. (d) is with $(n_1, n_2, n_3) = (50, 10, 40)$ and shows the converged solution. The potential energy (Equation 4.1) is mainly driven by the distance function d , which affects the level-set evolution. 143
- 4.6 The influence of noise on reconstructing three-fold circle with 200 points: (a)-(c) ALM and (d)-(f) SIM. The first column shows the reconstructed curves from clean data, and the second column the reconstructions from noisy data. The third column shows the comparison between the two reconstructed curves in first two columns. 144
- 4.7 Results by ALM with different r and ε . For each column, from top to bottom, $\varepsilon = 1, 1.5, 2$; and for each row, from left to right, $r = 0.5, 0.8, 1, 2$. Increasing ε renders the curve less sharp and more convex. Increasing r induces a stronger diffusion effect on ϕ^n 145
- 4.8 (a) Disc Q^n , (b) r_U^n , (c) r_L^n at certain iterations. (d) The region (in white) where d explicitly guides the level-set evolution by ALM. The distance function d refines the local structures and it is only active near $\{\phi^n = 0\}$. This partially explains the efficiency of ALM. 148
- 4.9 Effect of r_1 in ALM. For fixed $r_2 = 10, r_3 = 3$, and $\eta = 2$, increasing r_1 induces better reconstruction on the concave part. 170
- 4.10 Effect of r_2 in ALM. For fixed $r_1 = 10, r_3 = 3$, and $\eta = 2$, increasing r_2 induces better reconstruction on the concave part. 170

4.11	Effect of η in ALM. Here $r_1 = 15, r_2 = 10$, and $r_3 = 3$ are fixed. Increasing η induces reconstruction of the concave wedge. Although in cases, the energy curves are identical before the 100-th iteration, larger η suppresses the oscillation of the energy curve: yellow line ($\eta = 5$) is more stable compared to red ($\eta = 1$) or blue ($\eta = 0$).	171
4.12	(a) Results by ALM with noisy data and $r_1 = 15, r_2 = 10, r_3 = 3$. (b) shows a zoom-in of the right-bottom of (a). The noise is additive Gaussian with standard deviation 2. As η increases, the curve becomes less oscillatory.	172
4.13	With $\eta = 2$, comparison between OSM with $s = 2$ and ALM. In ALM, $r_1 = 15, r_2 = 10, r_3 = 3$ is used. Convergence to the steady state is faster for OSM. The reconstructed curves are shown in (b) for ALM and in (c) for OSM: ALM may shorten the curve, while OSM can extrude a corner to make a circular reconstruction.	172
4.14	(a) By OSM with $s = 2$ for the distance term in (Equation 4.27), the comparison between (I) $\eta = 0$ (green curve), (II) $s = 1$ (red curve), and (III) $s = 2$ (blue curve) for the curvature term. (b) OSM with $\eta = 2.5$ for $s = 2$ in the model (Equation 4.35). This is the blue curve in (a). OSM using $s = 2$ gives the best result in terms of capturing the structure of the underlying surface more accurately.	173
4.15	Comparison between the algorithm from [274] and OSM with or without curvature constraints: The first row results are by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. The second row results are by OSM without any curvature term, $\eta = 0$. The third row results are by OSM with curvature constraint ($s = 2$): (a) $\eta = 3$, (b) $\eta = 2$, (c) $\eta = 1$, and (d) $\eta = 2$. The shape of the underlying surface are more accurately captured using our proposed model with the curvature constraint.	174
4.16	Effect of η in OSM. (a) $\eta = 0$. (b) $\eta = 1$. (c) $\eta = 2$. (d) $\eta = 3$. (e) $\eta = 4$. As η increases, the two sharp corners are recovered better. As η gets larger, the corners get more circular to avoid large curvature.	175
4.17	By OSM, sparse data results with or without curvature constraints. (a) $\eta = 0$, and (b) $\eta = 4$ for a point cloud sampled from a Boomerang shape. (c) $\eta = 0$ and (d) $\eta = 2$, for a sparse square shape where only four corners and one point on each side are given. For both examples, with curvature constraint, the recovery is more accurate and sharper.	176
4.18	By OSM, extremely sparse data: (a) Given data. (b) The recovered result with $\eta = 1$ and (c) with $\eta = 1.5$. Even with extremely sparse data, curvature constraint model can reconstruct the square corners well.	176

4.19	By OSM, reconstruction with noisy data: (a) $\eta = 0$, (b) $\eta = 1$, (c) $\eta = 2$. The noise is Gaussian with standard deviation 1. As η gets larger, the two lower corners are better recovered.	177
4.20	Examples of three dimensional point cloud data. (a) A pyramid. (b) A yoyo. (c) An ice cream cone.	177
4.21	Reconstruction of the pyramid by OSM with $s = 2$: (a) Result with $\eta = 0$. (b) Result with $\eta = 10$. (c) Comparison of cross section along $y = 25$	178
4.22	Reconstruction of the ice cream cone by the algorithm from [274] and OSM with $s = 2$: (a) Result by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. (b) Result by OSM with $\eta = 0$. (c) Result by OSM with $\eta = 5$. (c) Comparison of cross sections of results by OSM along $y = 25$. .	179
4.23	Reconstruction of the ice cream cone by the algorithm from [274] and OSM with $s = 2$: (a) Result by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. (b) Result by OSM with $\eta = 5$. (c) Result by OSM with $\eta = 10$. (c) Comparison of cross sections of results by OSM along $y = 25$. .	180
5.1	(Top) Underwater image with heavy green cast. (Bottom) Result of the proposed method. In the middle, several zoomed-in regions are displayed for comparison. The resulted image has enhanced contrast, balanced colors, and many image contents, e.g., the patterns on the swimming shorts, are more recognizable. In this paper, all the underwater images are from the benchmark data set [321].	184
5.2	Pipeline of the proposed method. The result shown here uses $\eta = 10, \beta = 1/3$ as the model parameters.	187
5.3	(a) CIELAB gamut projection on the L^*-a^* plane. (b) Projection on the L^*-b^* plane. (c) Projection on the a^*-b^* plane. (d) Geometry for computing the chroma upper limit $C_{\max}^*(L, \theta + \gamma_i)$ in the direction of the hue angle $\theta + \gamma_i$ on the lightness level of L . Here the chroma direction falls within the sector $[\gamma_i, \gamma_{i+1}]$, $i = 1, 2, \dots$	190
5.4	(a) Gamma function used for chroma enhancement (Equation 5.13) with varying values of η . (b) Robust factor (Equation 5.14) for suppressing the noisy hues with varying values of β	194
5.5	Pre-processing by hue angle adjustment in the blue region of CIELAB. (a) Part of an underwater image. (b) Proposed method without the pre-processing (Equation 5.17). (c) Proposed method with the pre-processing. With the adjustment, the blueness on the strap is preserved.	196

5.6	Post-processing considering the HK-effect. (a) Zoom-in of part of the underwater image in Figure 5.1. (b) Proposed method without the post-processing (Equation 5.18). (c) Proposed method with the post-processing. Post-processing considering the HK-effect reduces over-exposure.	197
5.7	General examples of the proposed method. (a) A typical underwater image showing dominating blue cast, and the contrast is relatively low. (b) Result of the proposed method applied to (a) removes the blue cast and enhances the textures on the riverbed. (c) A blurry underwater image where objects are hardly visible. (d) Result of the proposed method applied to (c) which shows vibrant colors and sharp objects' boundaries. (e) A deep underwater image commonly seen in field exploration. (f) Result of the proposed method applied to (e) which renders the details of the structure of interest. .	198
5.8	The proposed method shows consistent performance for underwater images with different color casts. For each underwater image in the first 3 rows, we show their hue angle distributions in the second column; in the third columns we show the distribution of the final results, which are displayed in the last row. In all cases, we have $\eta = 6$ and $\beta = 1/4$	199
5.9	(a) Original underwater image. (b) Proposed method without applying the robust factor (Equation 5.14) ($\eta = 8$). (c) Proposed method without applying the robust factor ($\eta = 2$). (d) Proposed method with the robust factor ($\eta = 8, \beta = 1/3$). Using the robust factor suppresses the background noisy colors while enhancing the saturation of the other regions.	201
5.10	Effect of the chroma enhancement parameter η . (a) Original underwater image. (b) Result with $\eta = 2$. (c) $\eta = 4$. (d) $\eta = 10$. Here we fix $\beta = 1/4$	202
5.11	Qualitative comparison of different methods (a) Original underwater image. (b) Zhao et al. [352] (c) Peng et al. [319] (d) Histogram Equalization (HE) (e) Limare et al. [353] (f) Automatic Color Enhancement (ACE) [71, 354] (g) Local Color Correction [355] (h) Multiscale Retinex [329, 330] (i) Proposed Method.	205
5.12	Quantitative evaluation and comparison. We compare our methods with Histogram Equalization (HE), Peng et al. [319], and Automatic Color Enhancement (ACE) [71, 354]. The quality of each image is evaluated by UCIQE [317] (left, blue marks the best) and UIQM [356] (right, red marks the best). In all the cases, we use $\eta = 10$ and $\beta = 1/4$ for the proposed method.	206

- 6.1 The sensitivity of numerical differentiation to noise. (a) Graph of $\sin(x)$, $0 \leq x \leq 2\pi$ (black), and its noisy version (red) with Gaussian noise of mean 0 and standard deviation 0.01. (b) The first-order derivatives of the function (black) and the data (red). (c) The second-order derivatives of the function (black) and the data (red). The derivatives of data in (b) and (c) are computed using the five-point ENO scheme. As the order of derivative increases, the noise gets amplified. 213
- 6.2 Performance of SDD on the data in Figure 6.1. (a) Graph of $\sin(x)$, $0 \leq x \leq 2\pi$ (black) and the denoised data (red) using MLS. (b) First-order derivatives of the function (black) and the denoised data using SDD (red). (c) Second-order derivatives of the function (black) and the denoised data using SDD (red). Derivatives are computed by the five-point ENO scheme, and the smoothing operator S is MLS. 214
- 6.3 (a) and (b) illustrate the idea of TEE. (c) and (d) explain MTEE when $w = 2$. The blue arrows in (a) and (c) represent time evolution using the forward Euler scheme on a fine time grid with spacing $\widetilde{\Delta t} \ll \Delta t$. In (b), two different PDEs (green and red) are evolved, and the green one has a smaller TEE. In (d), the candidate PDEs are evolved from multiple time locations, and their numerical solutions are compared with the denoised data after a time length of $w\Delta t$ 219
- 6.4 An example of the ST iteration. Starting with a large number K , the first iteration gives rise to K candidate coefficients for $k = 1, \dots, K$. The PDE with the smallest MTEE is picked, e.g., SP(3) with cardinality $K_1 = 3$ and support \mathcal{A}_1 . The second iteration gives rise to the candidate coefficients only supported on \mathcal{A}_1 using SP(k) with $k = 1, 2, 3$. The PDE with the smallest MTEE is found, e.g., SP(2) with cardinality $K_2 = 2$ and support \mathcal{A}_2 . The third iteration does not change the support, i.e., $\mathcal{A}_3 = \mathcal{A}_2$, so the final output is the coefficient vector of SP(2). 220
- 6.5 Robustness of MTEE over TEE. (a) Noisy solution set of the Burgers' equation $u_t = -uu_x$ generated with $T = 0.05$, $\Delta t = 0.001$, $\Delta x = 1/256$. By evolving the noisy initial condition according to $u_t = -uu_x$, (b) shows the solution at $t = 0.02$ and (c) shows the solution at $t = 0.03$. The solution blows up at $t = 0.032$ 221
- 6.6 Noisy and denoised data of the transport equation (Equation 6.15), as well as simulations of the recovered PDE. (a) The clean data, (b) data with 10% noise, (c) the denoised data $S_x[U]$, (d) simulation of the PDE identified by ST and SC (identical). (e) Data with 30% noise, (f) the denoised data $S_{(x)}[U]$, and (g) simulation of the PDE identified by ST and SC (identical). 229

6.7	The average error e_c , e_r and e_e over 50 experiments for the transport equation (Equation 6.15) with respect to various noise levels. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviation is represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (c) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). The coefficient error e_c by ST and SC is significantly smaller than that of IDENT.	229
6.8	Robustness of SC to the choice of α for the recovery of the transport equation (Equation 6.15). (a) and (b) display e_c and e_r versus $1/\alpha$ respectively, with 1% (Blue), 5% (Red), 10% (Orange), 20% (Purple) noise. Each experiment is repeated 50 times, and the errors are averaged. We observe that SC is not sensitive to α , and there is a wide range of values for α that give rise to a small error.	230
6.9	Clean and noisy data of the transport equation (Equation 6.15) with the initial condition (Equation 6.16). (a) Clean data. (b) Noisy data with 10% noise. (c) Noisy data with 30% noise.	231
6.10	The average error e_c , e_r and e_e over 50 experiments for the Burgers' equation (Equation 6.17) with respect to various noise levels, where the initial condition is Equation 6.18. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviations are represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (c) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). The e_c , e_r and e_e of ST and SC are much smaller than those of IDENT.	233
6.11	The average error e_c , e_r and e_e over 50 experiments of the Burgers' equation with diffusion (Equation 6.20) with respect to various noise levels. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviations are represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (b) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). Among the three methods, ST gives the best result.	235
6.12	Robustness of SC to the choice of α for the recovery of the Burgers' equation with diffusion (Equation 6.20). (a) and (b) display e_c and e_r versus $1/\alpha$ respectively, with 0.5% (Blue), 1% (Red), 3% (Orange), 5% (Purple) noise. Each experiment is repeated 50 times, and the errors are averaged. When the noise level is low, such as 0.5% and 1%, there is a wide range of values for α , which give a small error. As the noise level increases, the range of the optimal α becomes narrow.	236

6.13	SDD results with different smoothers. The first row is the numerical solution of Equation 6.27 at $t = 0.15$ (0% noise) with the initial condition $u_0(x, y) = \sin(3\pi x) \sin(5\pi y)$ and its various partial derivatives. The second row shows the noisy data and its numerical derivatives when 5% Gaussian noise is added to the clean data. The bottom four rows are the SDD results at $t = 0.15$ using MA, cubic spline (CS), diffusion (DF), and MLS in order. While all methods recover U (the first row), the dynamics of the derivatives, especially in the third and fourth rows, are best preserved by MLS.	242
6.14	Probability of signed-support recovery $\mathbb{P}[\mathbb{S}_\pm(\hat{\beta}) = \mathbb{S}_\pm(\beta^*)]$ versus the grid size of temporal dimension N , and $\ \hat{z}_{\mathcal{S}^c}\ _\infty$ versus N are recorded on the same plot for Viscous Burger's equation in panel (a) and for KdV equation in panel (b), respectively.	261
6.15	Boxplots of $\ \hat{\mathcal{Q}}_N\ _\infty$ versus N are displayed for Viscous Burgers' equation in panel (a) and KdV equation in panel (b), respectively.	262
6.16	Left panel (a) displays the curves representing $\mathbb{P}[\mathbb{S}_\pm(\hat{\beta}) = \mathbb{S}_\pm(\beta^*)]$ versus N , when $\nu = 0.03, 0.02, 0.01, 0.005$. Right panel (b) exhibits the range of λ_N for which ℓ_1 -PsLS gives the solution $\hat{\beta}^\lambda$ such that $\mathcal{S}(\hat{\beta}^\lambda) \subseteq \mathcal{S}(\beta^*)$ with respect to N , when ν is set as 0.005.	263
7.1	Workflow of the proposed STIS model.	266
7.2	(a) Structure of a coordinate component of the TAC feature extractor \mathcal{N}_1 . The input is a TAC in \mathbb{R}^N ($N = 90$) and the output is a feature vector in $\mathbb{R}^{K'}$ ($K' = 32$). The convolution kernels are of size 3 and the zero-paddings are of size 1. The numbers above Conv are the number of channels, and those above Linear are node sizes. (b) Structure of the domain synthesizer \mathcal{N}_2 . The input is an image whose channels are TAC feature vectors. Here H is the image height and W is the image width such that $M = HW$. The output is an image with K channels, each of which defines a spatial basis for the reconstruction.	273
7.3	Interpretability of spatial and temporal bases identified by STIS using a low rank ($K = 3$). Each spatial basis roughly corresponds to homogeneous regions with similar dynamic features, and the associated temporal basis describes its contribution to the concentration distribution.	274
7.4	Performance on TAC reconstruction. (a) Three 3×3 square ROIs marked on the final frame of an original test sample. (b) Average TACs of the respective ROIs in (a).	275

7.5	Qualitative comparison of different methods on a test dPET image sequence. The mark region is further examined in Figure 7.6.	277
7.6	Zoom-in comparisons among (a) Original (b) SEMF7 (c) SEMF16 (d) Proposed (K3I2) (e) Proposed (K7I3) on the final frame of the tests in Figure 7.5.	278
7.7	Comparison of different methods on TAC reconstruction. (a) Three 3×3 square ROIs marked on the final frame of an original test sample. (b) Comparison of the average TACs in R1 (c) in R2, and (d) in R3.	278

SUMMARY

Patterns represent the spatial or temporal regularities intrinsic to various phenomena in nature, society, art, and science. From rigid ones with well-defined generative rules to flexible ones implied by unstructured data, patterns can be assigned to a spectrum. On one extreme, patterns are completely described by algebraic systems where each individual pattern is obtained by repeatedly applying simple operations on primitive elements. On the other extreme, patterns are perceived as visual or frequency regularities without any prior knowledge of the underlying mechanisms. In this thesis, we aim at demonstrating some mathematical techniques for representing patterns traversing the aforementioned spectrum, which leads to qualitative analysis of the patterns' properties and quantitative prediction of the modeled behaviors from various perspectives. We investigate lattice patterns from material science, shape patterns from computer graphics, submanifold patterns encountered in point cloud processing, color perception patterns applied in underwater image processing, dynamic patterns from spatial-temporal data, and low-rank patterns exploited in medical image reconstruction. For different patterns and based on their dependence on structured or unstructured data, we present suitable mathematical representations using techniques ranging from group theory to deep neural networks.

CHAPTER 1

INTRODUCTION

1.1 What is Pattern?

With the advance of data acquisition techniques and storage devices, the amount of accessible information grows exponentially, and it becomes more imperative than ever to identify the patterns buried in the ocean of data. *Pattern* [1, 2, 3, 4] is an abstract regularity discovered, derived, or sometimes invented to harness the correlations among individual events for the purpose of understanding, analysis, and prediction. Depending on the data types, patterns may be realized from diverse perspectives, and the established relations among samples can be either deterministic [1, 3] or stochastic [5, 6]. The formation of patterns is often causal [7], yet pattern recognition is mainly contingent on cognition [8, 9]. Successful pattern identification yields a considerable information reduction, which is generally based on the belief that subjects of interest can be embedded into certain low dimensional structures [10, 11]. However, these structures are almost always not unique due to the complexity nature of data. The superiority of one pattern representation over another is closely related to the specific applications.

1.1.1 Diverse Forms of Pattern

The most recognizable feature of pattern is visual repetition. For example, geometric patterns in architecture surface designs by tessellation [12] and garment textures [13] created by tactically translating or rotating certain primitive shapes. These patterns are *rigid* by careful design, such that the frequently occurring elements are identical and their distributions are completely predictable. There are more patterns in nature observable by human that do not contain rigorously repetitive units. Many echinoderm like starfish [14] and

crinoid [15] as well as snowflakes exhibit surprisingly diverse yet highly radial symmetries. Meandering rivers, crawling snakes, and brain corals all follow similarly looking sinuous paths. Romanesco broccoli, coastal boundary, frost crystals, and many others present a beautiful self-similarity, whose structures approximately keep repeating themselves when observed in increasingly smaller scales [16]. These *soft* patterns admit various flexible forms where both deterministic and stochastic factors contribute to the perceived regularity. The tolerance of lack of exact repetition is closely related to the cognitive process called *attention* [17]. This vital mechanism allocates human's limited cognitive processing resources to concentrate on specific aspects of information. It is striking that, although we are vividly submerged in the reality through our vision, more than 99% of the received visual data per second result in inattention blindness [18]. Consequently, when focusing on structural similarities, we are able to, or unconsciously forced to omit various kinds of discrepancies and extract the general regularities among separate parts or distinct objects.

Patterns are also identified via diverse stimuli ranging from sounds [19, 20, 21] and smells [22] to temperature [23] and electromagnetism [24]. The fate motif from the first movement of Beethoven's Symphony No.5 as well as numerous other great works [20] and the BACH motif (B flat-A-C-B natural) [21] employed by countless composers are well-known examples of frequently appearing phrase patterns rendering distinctive musical developments that tie the whole pieces into harmonic unities. The fact that gradually morphing from one odours to another induces noticeably different neuronal codings [22] implies a close correlation between the chemosensory signatures and olfactory pattern classification.

On more abstract levels, regularities studied in physics, chemistry, biology, linguistics, social structures, economics, political science, and history have greatly enriched the connotation of pattern. Patterns can be prescriptive such as grammar [25] or descriptive such as syntax [26]. Structural patterns describe static and stable relations among constituent components, e.g., infrastructure analysis [27], whereas dynamic patterns concentrate on

regularities along the temporal dimension, e.g., stock market analysis [28]. Long-term patterns study events which can extend to millions of years as discussed in Darwin's theory of evolution [29], and short-term patterns may focus on phenomena that only last seconds like neuronal firing [30]. There are also patterns of various scales such as macroeconomics [31] versus microeconomics [32], and theory of relativity [33] versus quantum mechanics [34].

1.1.2 Pattern Formation, Recognition, and Representation

Patterns can be studied mainly from three aspects: pattern formation, pattern recognition, and pattern representation. The first two characterize respectively the physical cause and psychological cause of diverse patterns, while the last one focuses on developing models that reproduce or approximate observed patterns.

Pattern formation characterizes the self-organizing mechanisms of complex systems from orderly generative rules and identifiable primitive elements. A complete description of pattern formation would allow exact reproducibility, which may be achieved in constructive manners or in experimental settings where all the determining factors are controllable. In most cases, only key causal factors are accessible, leaving the unidentified influences that yield insignificant variability as random variables. For instance, one of the driving forces of hexagonal patterns found among vegetation in flat terrains is ascribed to the positive water-biomass feedback between local vegetation growth and water transport towards the growth region [35]; meanwhile, such mechanism is also affected by disturbance regime, dominant plant species, and many other environmental factors [36]. Pattern formation in nature generally involves elements of multiple scales and distinctive properties [37, 38], which makes it demanding to develop a unified and comprehensive framework.

Although the cognitive processes are extremely sophisticated, pattern recognition in common experience is achieved effortlessly for human. There are two major challenges associated with pattern recognition. First, given that the consciousness of patterns emerges naturally for human, e.g., facial recognition [39], it is surprisingly difficult to develop an

artificial system that processes information on a level close to the way we perceive diverse stimuli. On the positive side, theories [40, 41] based on template matching, prototype matching, feature analysis, Fourier analysis have been proposed and successfully applied in different aspects of everyday life, and the cognitive processes modeled by these theories which we heuristically identify keep inspiring new technological developments. Second, there are countless patterns recognizable by human only if the information is converted to particular forms, and looking for the appropriate transformations is not always straightforward. Examples include line patterns in stellar spectra [42], ultrasonic patterns [43], acoustic fingerprint [44], and periodic patterns in biological sequences such as DNA [45].

Unlike pattern formation and recognition, in the studies of pattern representation, human takes a proactive role. For the same pattern, different types of representation can be devised from various perspectives. As an example, shape is a fundamental attribute of object besides other defining properties including color and texture [46]. Each shape is regarded as a unique pattern formed by certain combinations of visual cues, thus in this case, pattern formation is closely tied to its recognition. Generally, shape representations [47] either focus on contour or region, i.e., the boundary or the interior of a given shape, and each one of which is further classified based on whether the described features are structural or global. For instance, the chain code [48] is a contour-based structural method; the Fourier descriptor [49] is a contour-based global method; the convex hull is a region-based structure method; and the Zernike moment [50] is a region-based global method. we note that none of these approaches coincides with the complete cognitive process of shape recognition, yet they provide compact and useful information suitable for many purposes.

Specifically, qualitative pattern representation characterizes general structural features and provides critical insights for the underlying regularity; and quantitative pattern representation models the connection among samples, such that the pattern observed in the group is translated into numerical data. Moreover, a direct pattern representation is a simplification of the mechanisms behind pattern formation, and an indirect pattern representation

focuses on the distinguishing features of the pattern without heavily relying on the physical or psychological processes.

In most cases, pattern representation is only approximating or restricted to certain aspects of the associated pattern formation or recognition. However, sophisticated pattern representation establishes an accessible path to identifying and understanding dominant factors in pattern formation and recognition, which can also be efficiently utilized for artificial intelligence. For instance, the parse tree representation of the syntax pattern is frequently applied in machine translation [51], and various algorithmic descriptions of the Retinex theory [52, 53] instantiate the cognitive patterns in the human visual system, which have been extensively used to in computer vision [54, 55, 56]. More importantly, successful pattern representation sometimes predicts the existence of new patterns of groundbreaking significance. The most well-known example is the prediction of black hole from a peculiar solution of the Einstein field equations [57, 58] developed in 1915, and it was only recently unveiled to the public for the first time by the Event Horizon Telescope [59, 60] in 2019. In these works, mathematical models play indispensable roles which offer powerful analytic tools and quantifiable predictions. Not only can they precisely depicts the regularities embedded in diverse forms of pattern, but they also provide explicit and verifiable implications via rigorous derivations. Given their profound significance in numerous theories and countless applications, we will concentrate our attention to mathematical representations of pattern starting from the next section.

1.2 Mathematical Pattern Representation

Regarding sample attributes as variables and the governing pattern as operators relating them, mathematical pattern representation aims at developing mathematical models to characterize the pattern in a quantifiable manner based on tools such as groups, algebraic equations, dynamical systems, and probability distributions, etc. The modeled patterns include diverse phenomena discovered in nature and society, or they may well be found within

mathematics itself. Typically, a mathematical pattern representation is obtained by derivation starting from numerous axioms or principles of the underlying phenomena; this is called the model-based approach. As an alternative, the data-driven approach identifies the representation by learning features from a given dataset. There are also many hybrid methods which establish the basic framework using model-based ideas, while leaving certain parameters determined by the collection of data. Thanks to the soaring computing power, with their superior performances in countless applications, data-driven representations such as deep neural network (DNN) have received increasing popularity. Although many classically challenging problems have been addressed by data-driven approaches with satisfaction, it should be noted that model-based representations still play irreplaceable roles and can considerably improve the efficiency and accuracy of data-driven methods.

1.2.1 Model-based Approach

Traditionally, mathematical pattern representation involves defining equations and model assumptions. As an elementary example, assuming the heat conduction on a metal plate with diffusivity constant $\alpha > 0$ follows the Fourier's law [61], then the dynamic pattern of the heat variation within the plate's interior can be represented by the heat equation: $u_t = \alpha \Delta u$, where u is the temperature, $u_t = \partial u / \partial t$, the partial derivative of u with respect to time, measures the rate of change of temperature, and $\Delta = \partial^2 / \partial x^2 + \partial^2 / \partial y^2$ is the Laplacian operator in 2D Euclidean space. In addition, to complete the model, supplementary conditions are generally required. Continuing the previous example, geometry of the plate's boundary, initial heat distribution on the plate or the conduction behavior around the boundary, i.e., whether it is insulated or attached with external heat sources, are often specified to guarantee a well-posed problem in the Hadamard sense [62]: the system admits a unique solution which is continuously dependent on the given data. For many model-based representations, principles in physics are fundamental, and the properties of mathematical models are studied to understand the mechanisms behind pattern formation, e.g., equilib-

rium state [63], critical point [64], periodic orbit [65], bifurcation [66], and finite time blow-up [67]. Meanwhile, there are also many representations inspired by heuristics or guided by desired properties, which are usually encountered in models concerning pattern recognition, such as the Rudin-Osher-Fatemi (ROF) model [68, 69] for image restoration, Mumford-Shah’s functional [70] for segmentation, and perceptual color correction by Bertalmío et al. [71].

In addition to the analytic type described above, algebraic representation is particularly handy when describing symmetry, which is an elemental property rooting beneath every rigid pattern [72, 73]. Symmetry refers to a structural regularity of a system whose intrinsic properties remain invariant under a class of transformations. Focusing on the relations among basic units expressed via certain actions, algebraic structures such as groups, rings, and modules are powerful prototypes to precisely characterize the pattern and reveal hidden connections. For instance, all the repetitive planar patterns can be elegantly classified according to the family of wallpaper groups [74], and their 3D extensions are well described by layer groups [75], which are again special cases of the more general space groups [76] for the symmetries of 3D configurations. In these representations, every pattern corresponds to a group by identifying each group element with a basic spatial transformation, e.g., translation, and rotation, which keeps the pattern invariant; hence patterns are considered identical if and only if their group representations are isomorphic. Algebraic representations are also important in studying patterns within mathematics, and their applications permeate all branches. Gelfand rings in functional analysis [77], projective modules obtained by sections of vector bundles over smooth manifold [78], and Boolean algebra in probability theory [79] are a few examples among countless others. On a even more abstract level, the pattern among algebraic patterns is investigated in the category theory [80, 81], where objects and morphisms form the main elements to describe pattern formation. Therefore, in some sense, mathematics provides a universal language for pattern representation [82].

Other than analytic and algebraic ones, there are also mathematical pattern representations based on geometry, graph, and topology, etc., which are frequently employed in various applications. To reveal different properties of the underlying pattern, some representations are more advantageous than the others, and devising the appropriate models is often more of an art than science.

1.2.2 Data-driven Approach

Submerged in the massive amount of data from diverse sources, e.g., stock prices, photos, videos, customers' reviews, etc., we desperately need effective tools to organize and understand them. By doing so, we can achieve considerable reduction on the information to be stored, since the correlations among samples within the same class can be utilized to recover redundant data. Moreover, highly complex mechanisms of pattern formation and recognition can be discovered, which would be demanding to study for the conventional approaches due to non-linearity, discontinuities, and non-convexity.

Data-driven mathematical pattern representations aim at identifying the unknown mapping from given sample to its label. A sample usually encapsulates a bag of numeric or categorical attributes. When the label is continuous, the identification procedure is called regression; and when it is discrete, the procedure is known as classification. The mapping may belong to a finite dimensional functional space, thus it is determined by several unknown numbers, called parameters, and we call it a parametric problem. In contrast, when the model structure is not specified a priori, then it is non-parametric. The parameters as well as the functional relations in the case of non-parametric problems are obtained by learning features from collections of data. Specifically, when the dataset consists of sample-label pairs, it is a supervised learning; if the dataset only contains samples without access to any labels, it is a unsupervised learning. There are also notions like semi-supervised learning, reinforcement learning, and transfer learning which has been gaining more weights in recent researches. Among data-driven approaches, there are two closely related areas:

machine learning and data mining. While machine learning focuses on prediction through the pattern learned from the data, data mining aims at discovering new knowledge. Other than this discrepancy, they share most techniques.

Deep learning [83, 84] is a specific method of machine learning. It constructively approximates the unknown mapping from data to label via a consecutive composition of simple operations, such as affine transform, convolution, down-sampling, etc. Most importantly, the employment of nonlinear activation functions, e.g., sigmoid and rectified linear unit (ReLU), brings the necessary complexity for the powerful expressivity of the learned mapping. In diagram, the structure of such representations usually take forms of directed graphs, where each node stores a scalar, and each arrow from a source node to a target node indicates a simple mapping which is usually an affine transform followed an activation function. Such representations are thus called neural networks. In its most basic form, a neural network consists of three layers of nodes: the input layer where each node only has outward arrows; the output layer where each node only has inward arrows; and the hidden layer where nodes take data from the previous layer and send the processed data to the next layer. When there are multiple hidden layers stacked between the input and output layers, as what is implemented in practice, the structure is often called a deep neural network (DNN), and each layer of DNN often named after its main operations such as convolution layer and max-pool layer. Other popular architectures include fully convolutional neural network (FCNN) [85], U-net [86], recurrent neural network (RNN) [87], and deep residual network (ResNet) [88], etc.

Determining a neural network amounts to specifying the parameters of each layer involved, e.g., kernel weights for the convolution layers and matrix weights for linear layers, by optimizing a loss function L . In case of supervised learning, L compares the network outputs to the labels paired with the inputs; and in case of unsupervised learning, L evaluates the outputs in terms of certain desired properties. Due to the complexity of loss landscape [89], the optimization is almost always non-convex. Although the commonly applied

stochastic gradient descent methods help to escape local minima, the network training remains a delicate problem. In general, when the network is deeper, i.e., has more hidden layers, it acquires higher expressivity and approximates mappings with more complexity; meanwhile, the associated training easily gets unstable due to exploding gradient [90], dying neurons [91], and many other issues. For better convergence behaviors, much efforts have been made to design easily trained network architectures and more efficient training paradigms [92, 93, 94].

1.3 Organization of the Thesis

This thesis presents different types of mathematical pattern representation and their applications in various fields to display the elegance and effectiveness of both model-based and data-driven approaches. Through out the discussion, we make effort to illustrate the pros and cons of diverse mathematically equivalent representations in the context of specific applications. The main body of the thesis consists of two parts: Part I (chapter 2-5) focuses on model-based representations, and Part II (chapter 6 and 7) discusses data-driven methods. Finally, we summarize the thesis, comment on the comparison between model-based and data-driven methods, and discuss about future works in Chapter 8.

1.3.1 Contents of Part I (Chapter 2-5)

The first part of the thesis consists of Chapter 2-5 and discusses 4 different topics related to pattern representation. Each topic showcases a unique modeling technique with different mathematical flavor. The applications covered in this part ranges from material science to computer vision/graphics.

Overview of Chapter 2

In this chapter, we concentrate on one of the most fundamental rigid planar patterns, the regular 2D lattice, which admits a simple and natural representation via a finite dimensional

vector space. Although it describes lattice patterns exactly, such basic representation causes unexpected ambiguity for the purpose of classification. Alternatively, groups are perfect for characterizing the pattern symmetries, yet it is not suitable for quantifying the visual differences between lattice patterns. In this case, complex numbers serve a more powerful tool to express lattice patterns, and they elegantly encode geometric transformations via algebraic operations. Moreover, a complex quotient metric space, called the lattice metric space, can be developed where each lattice pattern corresponds to one and only one point, and visual differences among them are effectively described by distance in space. The rich geometric and topological structures naturally derived from the Poincaré metric and modular group theory empowers the lattice metric space to be a visually consistent system for lattice representation. In addition, we also discuss some applications of the lattice metric space in image processing and material sciences.

Overview of Chapter 3

In this chapter, shape patterns are mostly concerned. Different from rigid patterns, shapes are more versatile, and looking for a stable representation complying with visual perception allows diverse perspectives. We focus on representing the shape patterns using partial differential equations (PDEs). To represent binary shapes, or silhouette, in a compact format, we discuss the Hamilton-Jacobi skeleton region-based representation. It eliminates the redundant information by mapping the shape onto the singularity of the signed distance function determined by the shape boundary. Such lower-dimensional skeleton reveals homotopy types of raster shapes and allows accurate shape reconstruction. In addition, we discuss a contour based shape representation by approximating the polygonal boundaries of discrete shapes using Bézier polygons. This conversion from raster images to combinations of primitive elements specified by small numbers of points is known as vectorization. The vectorized results enjoy resolution-independent properties, which are important in font designs, logo editing, document transmission and preservation, etc. We describe

the application of affine-scale space induced by the affine shortening PDE, which allows to reduce pixel details irrelevant to shape recognition and to locate prominent curvature extrema as critical visual cues. We also include comparisons of the introduced method with state-in-art graphic software in terms of representation effectiveness both qualitatively and quantitatively.

Overview of Chapter 4

In this chapter, the subjects of interest are point clouds, which are assumed to be perturbed samples from a unknown submanifold of codimension one. We discuss some variaional techniques to represent the underlying submanifold pattern perceived from the configuration of points. In particular, each pattern implied by the given point cloud is represented by the level-set of a minimizer of certain Tikonov-type functional. Its geometric properties largely depend on the point-cloud distribution and the regularization explicitly defined via functionals of curvature or surface area. As the associated optimization problem is non-convex and has high-orders, straightforward implementation leads to slow convergence. We introduce fast-algorithms based on operator-splitting and semi-definite strategies to efficiently find the representation via alternative optimizing paradigms. With proper selections of parameters, these algorithms converge considerably faster than classical ones. Similar algorithmic ideas are crucial in numerically solving variational problems, and we will utilize them again in later chapters.

Overview of Chapter 5

In this chapter, we discuss pattern representation inspired by recognition and the related applications in underwater image color correction. Color constancy is an adaptive mechanism of human vision system (HVS), which is a vital property for relatively stable object recognition under varying illuminating conditions. Due to complex environmental factors, images acquired in underwater show saturated blue or green color cast, low contrast, or het-

erogeneous blurriness. Recovering the objects' colors in underwater scene is a challenging inverse problem, and the involved physics are rather complicated. Alternatively, we discuss several perceptually inspired methods focusing on reducing the visual degradation commonly faced in underwater images. In particular, we focus on introducing a simple least-square approach established in the CIELAB color space, where various interesting topics related to color representation will be covered.

1.3.2 Contents of Part II (Chapter 6-7)

In the second part of the thesis, we switch to data-driven pattern representations, where learning pattern features from collection of data is at the core. Although ideas from model-based approaches are often employed yielding great improvement on efficiency and accuracy, performance-related model parameters are determined from the given data. Here we focus on two interesting pattern representations constructed from spatial-temporal data.

Overview of Chapter 6

In this chapter, we discuss the problem of data-driven PDE learning, or infinite dimensional dynamic data mining. Conventionally, PDE models are established based on physical principles, and various assumptions are introduced to simplify the derivations, leading to the mathematical relations governing the essential dynamics. Data-driven approach offers a novel perspective, where PDEs are automatically determined by the experiment data without presuming the dominant dynamics or related differential operators, then we may discover the underlying physical laws by directly analyzing the PDEs. Different from classical PDE inverse problems that estimate unknown parameters of a fixed PDE or recent developments of PDE modeling based on DNN, here we focus on identifying the differential operators, either linear or nonlinear, solely based on the spatial and temporal characteristics of a single projection of the underlying PDE extracted by finite difference schemes. In addition to sparsity, we discuss some principles of model selection that help to identify the correct

PDE among numerous candidates. Due to the unboundedness of the candidate differential operators and noisy sample data, the identification can be very sensitive. We consider the importance of data denoising and describe an effective denoising technique, called Successive Denoising Differentiation (SDD). We include various experiments to show the model performance and validate the effectiveness.

Overview of Chapter 7

In this chapter, we take the deep learning perspective focusing on the restoration of dynamic positron emission tomography (dPET) from sparsely sampled time-dependent projection. In particular, we focus on an interesting architecture designed for spatial and temporal information synthesis (STIS). It takes advantage of the low rank structures of dPET image sequences, and alternatively enhances the spatial and temporal features of the underlying variation patterns of the radiotracer concentration. The classical end-to-end learning process is replaced by a hybrid paradigm which combines the interpretability of model-based approach and flexibility of data-driven model. With such design, the number of parameters for the DNN is greatly reduced, yielding an easier and more efficient training. Different from Chapter 6, where dynamic patterns are characterized by partial derivatives and differential equations, here the representation depends on the expressivity of DNNs. We will discuss the issue of hyper-parameter selections and compare STIS with some state-of-art DNN-based and model-based methods.

1.3.3 Reading Suggestions

This thesis contains many materials focusing on different aspects of mathematical pattern representation with diverse flavors and applications. The strategy of organizing them in the present order is as follows. From chapter 2 to chapter 7, the dependence of pattern representations on data becomes increasingly stronger, and the studied patterns are less rigid. This arrangement also reflects a transition from no parameters, to manually tuned parameters,

and finally to parameters determined by data. However, the reading does not have to follow this order. The materials for each topic are written in a self-contained manner as much as possible, and we restrain the overall comparison and comments to the final chapter. Since the essence of the thesis is to showcase the variety of interesting mathematical pattern representations across a broad range of applications, we would suggest the reader to at least briefly cover all the topics (without getting into the technical details) to fully appreciate the beauty and diversity of mathematical modeling techniques.

CHAPTER 2

SYMMETRIES AND METRIC STRUCTURES IN LATTICE PATTERNS

From material sciences to wallpaper pattern studies, there is a wide range of fruitful research on patterns both in theory and applications. Earlier studies [95, 96, 97] categorize patterns by symmetries, such as invariance under reflection or rotation. The frieze and wallpaper groups are applied in computer vision to identify periodic patterns [98]. Pattern recognition typically involves two closely related tasks: representation of regularities and automated classification [99]. Motivated by some of the current developments in material sciences [100, 101] and crystalline material image analysis [102, 103, 104, 105, 106, 107], we focus on the lattice pattern, which plays major roles in crystallography [108, 109], sampling theory [110], ecology [111] and many others. For example, the crystal structures (3D lattices) of halite (NaCl) and gold (Au) have distinct scales (NaCl constant: 5.640\AA [112]; Au constant: 4.065\AA [113]), which explains their proprietary differences. There is considerable research on detecting (non-superposed) 2D lattice patterns from images, e.g., using the peaks of the Fourier power spectrum to identify the lattice structure [114], and propagating an automatically suggested lattice pattern to the whole image by a tracking algorithm [115]. In [116], the authors associate the wallpaper groups with local affine transformations to cluster repeated elements, and Hays et al. [117] propose the higher-order affinities among potential texels to discover visually consistent lattices.

In this chapter, we introduce a framework to model and compare equivalent classes of lattices by constructing the **lattice metric space** \mathcal{L} equipped with a new metric $d_{\mathcal{L}}$ [118]. A clear definition of the equivalent lattice is a cornerstone to classification. In many context [119, 120, 121], a lattice is considered as an object that shares the structural features with \mathbb{Z}^n , $n \in \mathbb{N}_{\geq 1}$, thus all the n -dimensional lattices are equivalent. Focusing on symmetries, the theory of wallpaper groups [96] distinguishes five types of lattices: square, rect-

angular, hexagonal, rhombic, and parallelogrammic. Many works on grain images [104] use the lattice orientation to indicate distinct patterns. We define equivalent lattices to be identical lattices up to translation. The scale, as well as rotational differences, are concerned. From the positive minimal bases [122], we derive a new lattice representation using scale and shape descriptors defined on complex manifolds. This lattice space consists of equivalent classes of descriptors, which represent distinct lattice patterns up to translation. Building upon the Poincaré metric [123], we assign a natural metric structure to the lattice space. Then we will discuss two applications of lattice metric space. First, we will use the lattice metric as a quantification of the Lattice Identification and Separation Algorithm (**LISA**). It sequentially extracts lattice patterns from a superlattice image without any prior knowledge of the number of layers. Second, we apply the lattice metric space $(\mathcal{L}, d_{\mathcal{L}})$ to grain defect detection problem, and explore its further properties. As a general framework to describe and compare arbitrary lattice patterns, the lattice metric space is advantageous.

2.1 Preliminaries and Notations

A typical definition of lattice starts from two linearly independent vectors, b_1 and b_2 , as a basis. A lattice is a set of linear combination of the basis vectors with integer coefficients. In the two dimensional space, we utilize the complex notation, $b_j = x_j + iy_j \in \mathbb{C}$, $x_j, y_j \in \mathbb{R}$, $j = 1, 2$, for simplicity.

Definition 2.1.1 (2D Lattice, Basis). *Given a pair of complex numbers $(b_1, b_2) \in \mathbb{C}^2$ satisfying $b_1 \neq 0$ and $\text{Im}(b_2/b_1) \neq 0$, a 2D lattice determined by (b_1, b_2) is defined as the set:*

$$\Lambda(b_1, b_2) = \{k_1 b_1 + k_2 b_2 \mid k_1, k_2 \in \mathbb{Z}\},$$

and the pair (b_1, b_2) is called a basis for $\Lambda(b_1, b_2)$.

The condition $\text{Im}(b_2/b_1) \neq 0$ is equivalent to two vectors b_1 and b_2 being linearly

independent. For any lattice $\Lambda(b_1, b_2)$, by reordering or multiplying -1 if necessary, we can have $|b_1| \leq |b_2|$ and $\text{Im}(b_2/b_1) > 0$, i.e., the basis (b_1, b_2) is positive. In the rest of this chapter, we assume that a basis is positive, unless stated otherwise. The key to distinguishing lattices depends on the precise definition of equivalent bases.

Definition 2.1.2 (Equivalent Bases). *Let (b_1, b_2) and $(b'_1, b'_2) \in \mathbb{C}^2$. If $\Lambda(b_1, b_2) = \Lambda(b'_1, b'_2)$, then (b_1, b_2) and (b'_1, b'_2) are called a pair of equivalent bases for $\Lambda(b_1, b_2)$.*

Given two bases (b_1, b_2) and (b'_1, b'_2) , they are equivalent if and only if the matrix

$$A = \begin{bmatrix} \text{Re}(b'_1) & \text{Im}(b'_1) \\ \text{Re}(b'_2) & \text{Im}(b'_2) \end{bmatrix} \begin{bmatrix} \text{Re}(b_1) & \text{Im}(b_1) \\ \text{Re}(b_2) & \text{Im}(b_2) \end{bmatrix}^{-1}$$

has integer entries and its determinant is ± 1 . The definition of lattice using linearly independent vectors is natural and intuitive, yet it lacks a clear way to define equivalence classes that offers a simple measure for the lattice comparison.

Another important notion is the **minimal basis** [124]. A lattice basis (b_1, b_2) is minimal if $\max(|b_1|, |b_2|) \leq |b_1 \pm b_2|$. For a positive basis (b_1, b_2) , it is minimal if and only if $|b_2/b_1| \geq 1$ and $0 \leq \text{Re}(b_2/b_1) \leq 1/2$. Moreover, it can be efficiently transformed to an equivalent minimal basis using the Positive Gauss reduction algorithm [122]. While $|b_2| < |b_1|$, repeat the following until stabilization: $(b_1, b_2) = (b_2, -b_1)$, $q = \lfloor \text{Re}(b_1/b_2) \rfloor$, here $\lfloor x \rfloor$ denotes the closest integer to $x \in \mathbb{R}$, and $b_2 = b_2 - qb_1$. Figure 2.1 demonstrates three different bases generating an identical lattice and (a) $(3, 4i)$ has the shortest components among all the equivalent bases.

We assume that the given image $U : \mathbb{R}^2 \rightarrow [0, 1]$ contains N lattices $\{\mathcal{T}_{\mu_j} \Lambda(b_{j,1}, b_{j,2})\}_{j=1}^N$:

$$U = \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda(b_{j,1}, b_{j,2}) + R, \quad (2.1)$$

where $\mathcal{T}_{\mu} \Lambda(b_1, b_2)$ denotes a lattice image translated from 0 by $\mu \in \mathbb{C}$, and R is the residual image. For the purpose of visualization, we put a point spread function (PSF) of Gaussian

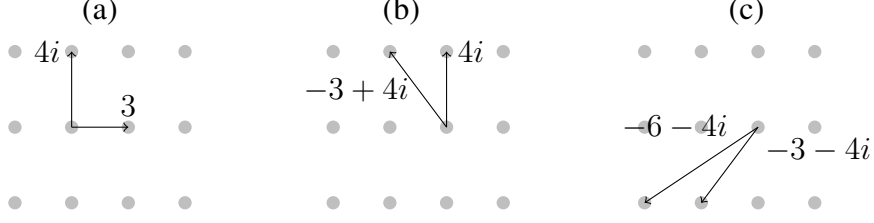


Figure 2.1: Equivalent bases and the minimal basis. (a) $\Lambda(3, 4i)$, (b) $\Lambda(4i, -3 + 4i)$, and (c) $\Lambda(-3 - 4i, -6 - 4i)$ represent an identical lattice. (a) $(3, 4i)$ is a minimal basis: $|\operatorname{Re}(\frac{4i}{3})| = 0 < \frac{1}{2}$. (b) $(4i, -3 + 4i)$ is not minimal: $|\operatorname{Re}(\frac{-3+4i}{4i})| = 1 > \frac{1}{2}$, and (c) $(-3 - 4i, -6 - 4i)$ is not positive: $\operatorname{Im}(\frac{-6-4i}{-3-4i}) = -\frac{12}{25} < 0$.

G_σ with standard deviation σ to each lattice point location [125]. The given image can be understood as each lattice impulse being convolved with a G_σ ; that is,

$$[\mathcal{T}_\mu \Lambda(b_1, b_2)](x, y) = \sum_{z \in \Lambda(b_1, b_2) + \mu} G_\sigma * \delta(z - x - iy), \quad (x, y) \in \mathbb{R}^2.$$

Here $\Lambda(b_1, b_2) + \mu = \{z + \mu \mid z \in \Lambda(b_1, b_2)\}$ is the set of translated lattice points. δ is the Dirac delta function in a distributional sense, i.e., a linear functional that evaluates a function at 0: $\delta[f] = \int_{-\infty}^{\infty} f(x)\delta(x) dx = f(0)$ for any function f defined on \mathbb{R} . Moreover, all the visible particles are assumed to be homogeneous. Even if multiple lattice points overlapping at the same location, the intensities are bounded by 1. This condition is ensured by the normalization in section 2.5. We note that for different atom configurations (such as different color atoms or different shapes), as long as one can roughly identify the location of each atoms with appropriate preprocessing, this method can be applied.

To capture the periodicities of a lattice pattern, we utilize the Fourier and Radon transforms. The arguments of the basis vectors are important features for the lattice identification. In the Cartesian coordinate, for an arbitrary point (ξ, ν) in the frequency domain, let $\theta = \tan^{-1}(\nu/\xi)$ denote its argument; using Taylor expansion, we see that the estimation error in the argument $|\Delta\theta| \approx \left| \frac{\xi\Delta\nu - \nu\Delta\xi}{\xi^2 + \nu^2} \right|$ depends on ξ and ν . That is, to control $\Delta\theta$, the spatial grid size, $\Delta\xi$ and $\Delta\nu$, must vary according to the location of the point. Compared

to the Cartesian coordinate, the polar coordinate is more effective. Hence, we exploit the Fourier Slice Theorem [126] to compute the 2D Fourier transform on the polar coordinate.

Theorem 2.1.1 (Fourier Slice Theorem). *Consider a function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, and denote \hat{f} as the Fourier transform, then:*

$$\hat{f}(\gamma \cos \alpha, \gamma \sin \alpha) = \widehat{\mathcal{R}_\alpha[f]}(\gamma), \text{ for any } \gamma \in \mathbb{R}, \alpha \in [0, \pi),$$

where $\mathcal{R}_\alpha[f](\gamma) = \mathcal{R}[f](\gamma, \alpha)$, and $\mathcal{R}[f]$ is the Radon transform of f defined by:

$$\mathcal{R}[f](\gamma, \alpha) = \int_{-\infty}^{+\infty} f(\gamma \cos \alpha - t \sin \alpha, \gamma \sin \alpha + t \cos \alpha) dt, \gamma \in \mathbb{R}, \alpha \in [0, \pi).$$

To construct a metric space, we review the following concepts [127].

Definition 2.1.3 (Quotient pseudometric). *Suppose (X, D) is a metric space, and \sim is an equivalence relation defined on X . Then the quotient pseudometric \overline{D} for X/\sim is defined as:*

$$\overline{D}([x], [y]) = \inf \{ D(p_1, q_1) + \cdots + D(p_n, q_n) \},$$

where \inf is taken over all finite sequences p_1, \dots, p_n and q_1, \dots, q_n in X such that $[p_1] = [x]$, $[q_n] = [y]$ and $[p_{i+1}] = [q_i]$, $i = 1, 2, \dots, n-1$.

The spaces we consider are Hausdorff spaces, i.e., for every pair of distinct points, each one of them has a neighborhood not containing the other. Consequently, all the quotient pseudometrics in this paper are metrics.

Definition 2.1.4 (Product Metric). *Suppose $(X_1, d_1), \dots, (X_n, d_n)$ are metric spaces, and D is an Euclidean norm on \mathbb{R}^n , then the product metric D_{d_1, \dots, d_n} associated with d_1, \dots, d_n*

for the space $X_1 \times \cdots \times X_n$ is defined as:

$$D_{d_1, \dots, d_n}((x_1, \dots, x_n), (y_1, \dots, y_n)) = D((d_1(x_1, y_1), \dots, d_n(x_n, y_n))) .$$

Remark 2.1.1. *The formal definition of the minimal basis that involves successive minima can be found in [124]. We note that the minimal basis is a special case of the notion of reduced basis. Variants of the reduced basis include the well-known Minkowski reduced basis [128, 129], the generalized Gauss-reduced basis [130], the Hermite-Korkine-Zolotarev reduced basis [131, 132], and the Lenstra-Lenstra-Lovász reduced basis [133]. They consider different relaxations, since finding the shortest vector using L_2 -norm is NP-hard for randomized reductions [134].*

Remark 2.1.2. *Vallée and Vera [122] include discussions about acute bases, which are characterized by $\text{Re}(b_2/b_1) \geq 0$. If (b_1, b_2) is a positive basis, then the orientation is guaranteed, but it is not necessary that b_1 and b_2 have an acute angle. In this paper, we prioritize the orientability, thus we choose to focus on positive bases.*

2.2 Lattice Feature Descriptors β and ρ

We introduce the representation for a lattice using a pair of complex numbers $(\beta, \rho) \in \mathbb{C}^2$, which we call descriptors [118]. They are derived from the positive minimal bases [122]. The key idea is that a lattice can be realized by transforming a unit lattice, $\Lambda(1, i)$. For instance, stretching or shrinking $\Lambda(1, i)$ along the direction of i gives rectangular lattices; and rotating i gives different lattice patterns, including the hexagonal lattice. One of the advantages of descriptors is that, compared to the Definition 2.1.2, the number of equivalent representations reduces considerably from infinite to only a few. By exploiting the modular group theory [135], we can fully characterize these equivalence relations. Descriptors modulo these relations are used as elements for the lattice space constructed in section 2.3.

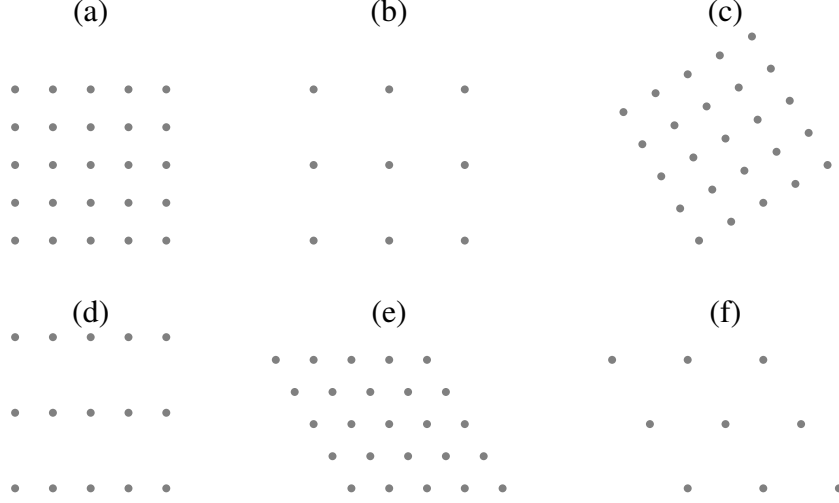


Figure 2.2: Effects of changing β and ρ . (a) $\Lambda\langle 1, i \rangle$, (b) $\Lambda\langle 2, i \rangle$, (c) $\Lambda\langle e^{i\pi/6}, i \rangle$, (d) $\Lambda\langle 1, 2i \rangle$, (e) $\Lambda\langle 1, e^{2\pi i/3} \rangle$, and (f) $\Lambda\langle 2, e^{2\pi i/3} \rangle$. From (a) to (b), β is changed from 1 to 2. From (a) to (c), β is rotated. From (a) to (d), ρ is changed from i to $2i$. From (a) to (e), ρ is rotated. From (a) to (f), both β and ρ are changed.

Definition 2.2.1 (Scale and Shape Descriptor). *Given a lattice $\Lambda(b_1, b_2)$ where (b_1, b_2) is a positive minimal basis, we define:*

Scale descriptor: $\beta = b_1$;

Shape descriptor: $\rho = b_2/b_1$.

We denote $\Lambda\langle \beta, \rho \rangle$ to be a lattice spanned by β and $\beta\rho$, i.e., $\Lambda\langle \beta, \rho \rangle = \Lambda(\beta, \beta\rho)$.

Figure 2.2 illustrates various effects of changing β and ρ . From (a) to (b), only β is changed from 1 to 2, and from (a) to (c), β is rotated. From (a) to (d) ρ is changed from i to $2i$, and from (a) to (e), ρ is rotated. From (a) to (f) both β and ρ are changed. Varying the scale descriptor β corresponds to zooming and rotating while changing the shape descriptor ρ leads to sheering, elongating, and shrinking asymmetrically. Algebraically, a pair of equivalent bases determines a simple relation between the associated descriptors.

Proposition 2.2.1 (Necessary condition). *If two lattices $\Lambda\langle \beta, \rho \rangle$ and $\Lambda\langle \beta', \rho' \rangle$ are equivalent, then there exists $k_i \in \mathbb{Z}$, $i = 1, 2, 3, 4$ with $k_1k_4 - k_2k_3 = 1$, such that the following*

hold:

$$\beta' = e^{i\text{Arg}(k_1+k_2\rho)}\beta, \text{ and} \quad (2.2)$$

$$\rho' = (k_3 + k_4\rho)/(k_1 + k_2\rho). \quad (2.3)$$

Proof. Note that $\Lambda\langle\beta, \rho\rangle = \Lambda\langle\beta', \rho'\rangle$ if and only if there is a unimodular matrix $U = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix}$, $k_i \in \mathbb{Z}$, $i = 1, 2, 3, 4$, such that $U \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, where $b_1 = \beta$, $b_2 = \beta\rho$, $b'_1 = \beta'$ and $b'_2 = \beta'\rho'$ are the associated positive minimal bases, respectively. From the matrix multiplication, Equation 2.3 follows immediately. Because the bases are minimal, $|b_1| = |b'_1|$ implies $b'_1 = e^{i\theta}b_1$ for some $\theta \in [0, 2\pi]$. Combining this with $b'_1 = k_1b_1 + k_2b_2$ gives $k_1 + k_2\rho = e^{i\theta}$, thus $\theta = \text{Arg}(k_1 + k_2\rho)$ and Equation 2.2 follows. In addition, since (b_1, b_2) and (b'_1, b'_2) are positive, $\det U = k_1k_4 - k_2k_3 = 1$. \square

In subsection 2.2.2, we apply the modular group theory to prove the converse of Proposition 2.2.1; hence, whether two descriptors generate an identical lattice can be easily determined. As a preparation, we state the following lemma.

Lemma 2.2.1. *The converse of Proposition 2.2.1 holds if*

$$|k_1 + k_2\rho| = 1.$$

Proof. Denote $c = \frac{1}{|k_1+k_2\rho|} = \frac{e^{i\text{Arg}(k_1+k_2\rho)}}{k_1+k_2\rho}$, then from Equation 2.2, we have $b'_1 = \beta' = c(k_1 + k_2\rho)\beta = c(k_1b_1 + k_2b_2)$. Notice that Equation 2.3 reads $b'_2 = b'_1 \frac{k_3+k_4\rho}{k_1+k_2\rho} = c(k_1 + k_2\rho)\beta \frac{k_3+k_4\rho}{k_1+k_2\rho} = c(k_3b_1 + k_4b_2)$. The lemma is thus proved. \square

Since the condition Equation 2.2 shows the dependency on ρ , we start with relations between shape descriptors specified by Equation 2.3.

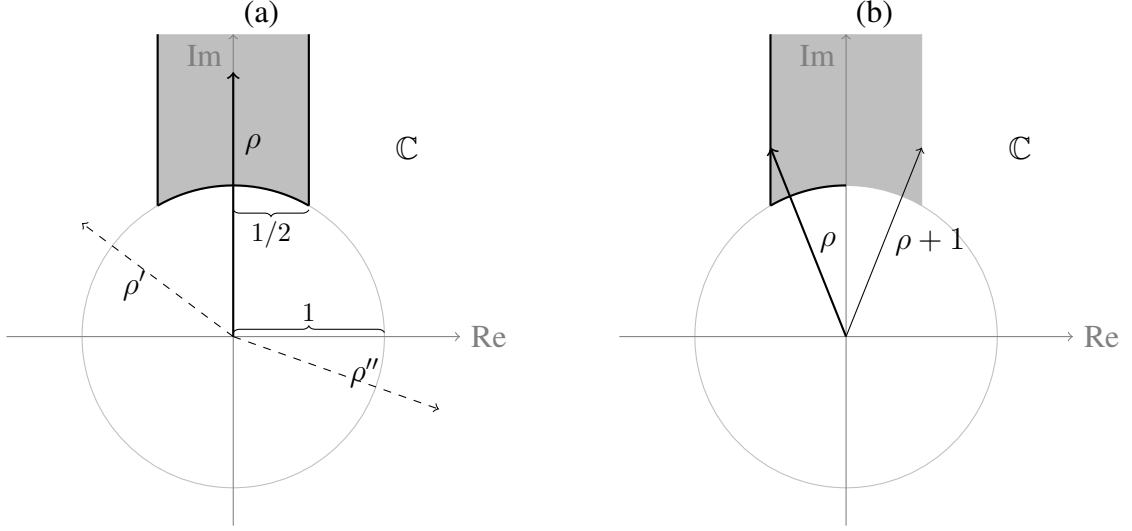


Figure 2.3: Region \mathcal{P} and the fundamental set of Γ -actions. (a) \mathcal{P} is the gray region including the boundary represented in \mathbb{C} . ρ , ρ' and ρ'' are the shape descriptors for $\Lambda(3, 4i)$, $\Lambda(4i, -3 + 4i)$, and $\Lambda(-3 - 4i, -6 - 4i)$ respectively from in Figure 2.1. Since $4i/3 \in \mathcal{P}$, $(3, 4i)$ is a positive minimal basis. (b) A fundamental set of the modular group Γ acting on the upper half plane. If $\text{Re}(\rho) = -1/2$, ρ and $\rho + 1$ are in the same orbit of a Γ -action.

2.2.1 Equivalence Classes of Shape Descriptor ρ

Using basic geometry, it is straightforward to show that the definition of the positive minimal basis is equivalent to the ratio $\rho = b_2/b_1$ belonging to the following region:

$$\mathcal{P} = \{z \in \mathbb{C} \mid |z| \geq 1, |\text{Re}(z)| \leq \frac{1}{2}, \text{Im}(z) > 0\} \subset \mathbb{C}. \quad (2.4)$$

Figure 2.3 (a) shows \mathcal{P} as the gray region including the boundary. We draw the shape descriptors ρ , ρ' and ρ'' for $\Lambda(3, 4i)$, $\Lambda(4i, -3 + 4i)$, and $\Lambda(-3 - 4i, -6 - 4i)$ respectively from Figure 2.1. All of them represent the same lattice, but only $(3, 4i)$ is a positive minimal basis since its shape descriptor $4i/3$ is in \mathcal{P} .

Equation 2.3 can be viewed as a transformation defined on the upper-half plane $\mathcal{H} =$

$\{z \mid \text{Im}(z) > 0\}$ restricted to \mathcal{P} , which can be expressed as:

$$z \mapsto \frac{k_3 + k_4 z}{k_1 + k_2 z}, \{k_i\}_{i=1}^4 \subset \mathbb{Z}, \text{ such that } k_1 k_4 - k_2 k_3 = 1, \text{ for any } z \in \mathcal{H}. \quad (2.5)$$

Each map of this form is a special case of the Möbius transformations, and the set of these transformations with the function composition gives the well-known modular group [135], denoted by Γ . The elements of Γ naturally act on \mathcal{H} as specified in Equation 2.5. Within the context of group actions, the condition Equation 2.3 characterizes an equivalence relation among shape descriptors, i.e., two shape descriptors are **equivalent** if they satisfy Equation 2.3.

The modular group Γ reveals the significance of the region \mathcal{P} defined in Equation 2.4. Notice that \mathcal{P} minus half of its boundary, i.e., the set:

$$\mathcal{P} \setminus \left(\{z \in \mathcal{H} \mid \text{Re}(z) = \frac{1}{2}\} \cup \{z \in \mathcal{H} \mid 0 < \text{Re}(z) < \frac{1}{2}, |z| = 1\} \right), \quad (2.6)$$

is a fundamental set for the Γ -actions [135], see Figure 2.3 (b). Every element in the fundamental set is a representative of one and only one orbit, and every orbit corresponds to a unique representative. No two shape descriptors in Equation 2.6 are equivalent. This provides a key insight that equivalent shape descriptors only occur on the boundary of \mathcal{P} .

Following the approach of Alperin on the modular group [136], we enumerate all the classes of equivalent shape descriptors systematically. Any Γ -action is a composition of a finite sequence of two basic transformations: translation T and inversion followed by reflection S respectively defined as

$$T : z \mapsto z + 1, \text{ and } S : z \mapsto -1/z, \text{ for any } z \in \mathcal{H}.$$

Any element in Γ can be written as $S^{k_1} T^{l_1} S^{k_2} T^{l_2} \dots S^{k_m} T^{l_m}$ for some $k_j \in \{0, 1\}$, $l_j \in \mathbb{Z}$, and $j = 1, 2, \dots, m$, where $m \in \mathbb{N}$. Focusing on the Γ -actions expressed as sequences of

S and T whose images have non-empty intersection with \mathcal{P} , we arrive at a full characterization of the equivalence classes of shape descriptors.

Proposition 2.2.2. *Given a shape descriptor $\rho \in \mathcal{P}$, based on its location, we list all the shape descriptors equivalent to it as follows:*

Location of ρ	All the equivalent shape descriptors
$\{z \in \mathcal{P} \mid z > 1, \operatorname{Re}(z) < 1/2\}$	ρ
$\{z \in \mathcal{P} \mid \operatorname{Re}(z) = -1/2, z > 1\}$	$\rho, T\rho$
$\{z \in \mathcal{P} \mid \operatorname{Re}(z) = 1/2, z > 1\}$	$\rho, T^{-1}\rho$
$\{z \in \mathcal{P} \mid z = 1, 0 \leq \operatorname{Re}(z) < 1/2\}$	$\rho, S\rho$
$e^{i2\pi/3}$	$\rho, S\rho, T\rho, T^{-1}S\rho, ST\rho, TST\rho$
$e^{i\pi/3}$	$\rho, S\rho, T^{-1}\rho, TS\rho, ST^{-1}\rho, STS\rho$

For any $\rho \in \mathcal{P}$, the number of equivalent shape descriptors is equal to the number of fundamental regions having ρ as a common point [135]; hence, the list of equivalent shape descriptors is complete. Geometrically, the small sizes of equivalence classes come from the restriction that both ρ and ρ' belong to \mathcal{P} . In effect, the principle behind the reduction is the uniqueness of successive minima of a finite dimensional lattice. This requires that the transformations must preserve norms, and they form a proper subset of the modular group.

Remark 2.2.1. *The notion of shape descriptor ρ is compatible with the five classes of lattices in the wallpaper groups [96]. For a lattice $\Lambda\langle\beta, \rho\rangle$, if $\rho = \pm\frac{1}{2} + i\frac{\sqrt{3}}{2}$, then it is hexagonal; if $\rho = i$, then it is square; if $\operatorname{Re}(\rho) = 0$, then it is rectangular; if $|\operatorname{Re}(\rho)| = \frac{1}{2}$ or $|\rho| = 1$, then it is rhombic; otherwise, it is parallelogrammic. The shape descriptor ρ recognizes finer differences, and with the scale descriptor β , they represent all lattice patterns up to translation.*

2.2.2 Equivalence Conditions for Scale Descriptors

Equation 2.2 for scale descriptors shows dependency on the equivalence relations between shape descriptors. The choice of the Γ -action that achieves an equivalence relation between

ρ and ρ' restricts the angles between β and the set of scale descriptors satisfying Equation 2.2 with β . Every Γ -action is associated with a matrix $\begin{bmatrix} k_4 & k_3 \\ k_2 & k_1 \end{bmatrix}$ up to sign, whose entries in the first row are the coefficients in the numerator in Equation 2.3, and those in the second row are the coefficients in the denominator. The composition of the actions is equivalent to the matrix multiplication of the associated matrices. For the nontrivial actions in Proposition 2.2.2, their matrix representations are:

$$\begin{aligned} T &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad T^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}, \quad S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad T^{-1}S = \begin{bmatrix} -1 & -1 \\ 1 & 0 \end{bmatrix}, \quad TS = \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix}, \\ TST &= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad ST = \begin{bmatrix} 0 & -1 \\ 1 & 1 \end{bmatrix}, \quad ST^{-1} = \begin{bmatrix} 0 & -1 \\ 1 & -1 \end{bmatrix}, \quad STS = \begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix}. \end{aligned}$$

This entire list of possible Γ -actions, that relate equivalent shape descriptors, contains critical information. First, for any $\rho \in \mathcal{P}$, the corresponding Γ -actions in Proposition 2.2.2 always satisfy $|k_1 + k_2\rho| = 1$, where the action is expressed as a matrix $\begin{bmatrix} k_4 & k_3 \\ k_2 & k_1 \end{bmatrix}$. Therefore, combining Proposition Equation C.23 and Lemma 2.2.1 yields our fundamental result.

Theorem 2.2.1 (Equivalent descriptors). *Two lattices $\Lambda\langle\beta, \rho\rangle$ and $\Lambda\langle\beta', \rho'\rangle$ are equivalent if and only if there exists $k_i \in \mathbb{Z}$, $i = 1, 2, 3, 4$ with $k_1k_4 - k_2k_3 = 1$, such that the following hold:*

$$\begin{aligned} \beta' &= e^{i\text{Arg}(k_1 + k_2\rho)}\beta, \quad \text{and} \\ \rho' &= (k_3 + k_4\rho)/(k_1 + k_2\rho). \end{aligned}$$

Second, the list of matrix representations allows us to summarize all the variants of Equation 2.2.

Proposition 2.2.3. *Given a scale descriptor $\beta \in \mathbb{C} \setminus \{0\}$ and two shape descriptors $\rho, \rho' \in$*

Γ -actions	Condition (Equation 2.2) satisfied with
I, T, T^{-1}	$\pm\beta$
$S, T^{-1}S, TS$	$\exp(i\text{Arg}(\rho))\beta$
TST, ST	$\exp(\pm i\text{Arg}(1 + \rho))\beta$
ST^{-1}, STS	$\exp(\pm i\text{Arg}(1 - \rho))\beta$

\mathcal{P} . If ρ and ρ' are equivalent using the Γ -actions in the left column of the following table, then the scale descriptors that satisfy the condition Equation 2.2 with β are listed in the right column correspondingly.

2.3 From Descriptors to Lattice Metric Space $(\mathcal{L}, d_{\mathcal{L}})$

2.3.1 Definition of Lattice Metric Space

Using the descriptors, we present the lattice space \mathcal{L} equipped with a metric $d_{\mathcal{L}}$. The equivalence relations discussed in section 2.2 allow every lattice pattern to be uniquely represented by a point in this space.

Definition 2.3.1 (Lattice Space). *Let \mathcal{P} be the set of shape descriptors ρ (Equation 2.4), and $\mathcal{K} = \mathbb{C} \setminus \{0\}$ be the set of scale descriptors β . With the induced topology, the lattice space \mathcal{L} is defined as*

$$\mathcal{L} = (\mathcal{K} / \sim_1 \times \mathcal{P} / \sim_2) / \sim_3, \quad (2.7)$$

where the three equivalence relations are:

1. $\beta \sim_1 -\beta$, for any $\beta \in \mathcal{K}$, i.e., $\Lambda\langle\beta, \rho\rangle = \Lambda\langle-\beta, \rho\rangle$.
2. $\rho \sim_2 \rho'$, for any $\rho, \rho' \in \mathcal{P}$ with $\text{Im}(\rho) = \text{Im}(\rho')$ and $|\text{Re}(\rho)| = |\text{Re}(\rho')| = 1/2$, i.e., $\Lambda\langle\beta, \rho\rangle = \Lambda\langle\beta, \rho'\rangle$.
3. $\langle[\beta]_1, [\rho]_2\rangle \sim_3 \langle[\beta\rho]_1, [-1/\rho]_2\rangle$, for any $\beta \in \mathcal{K}$, and $\rho \in \mathcal{P}$ with $|\rho| = 1$, i.e., $\Lambda\langle\beta, \rho\rangle = \Lambda\langle\beta\rho, -1/\rho\rangle$.

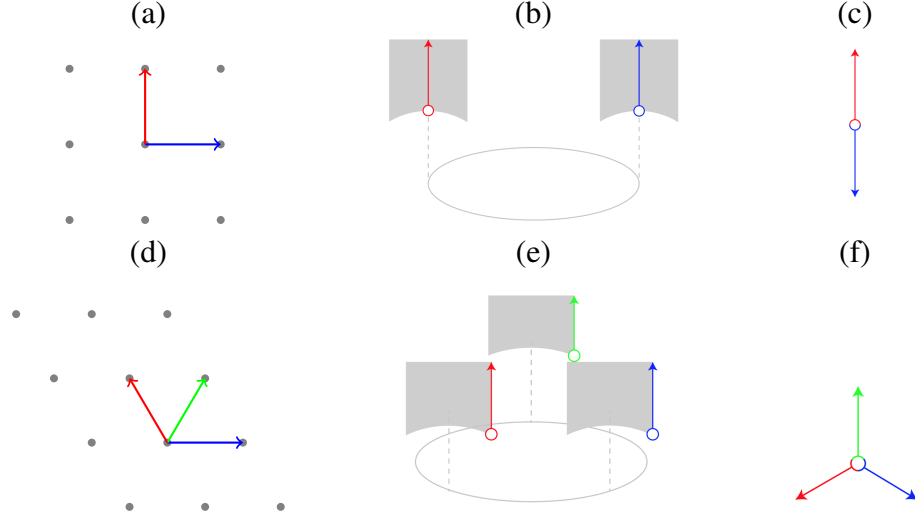


Figure 2.4: Examples of subspaces of \mathcal{L} . For any $\beta \in \mathcal{K}$, (a) shows a square lattice $\Lambda\langle\beta, i\rangle$. The red and blue arrows indicate two directions. Stretching $\Lambda\langle\beta, i\rangle$ along them represents two different families of lattices. They form a subspace of \mathcal{L} shown in (b), which is homeomorphic to \mathbb{R} as in (c). (d) shows a hexagonal lattice $\Lambda\langle\beta, e^{i\pi/3}\rangle$. Stretching it along the three marked directions generates three distinct families of lattices. (e) is the subspace they form in \mathcal{L} , which is homeomorphic to the structure in (f).

We denote $[\beta, \rho]$ as an element in \mathcal{L} considering the equivalence relations.

The quotient space \mathcal{K}/\sim_1 consists of scale descriptors β up to sign, which amounts to only considering the upper-half plane \mathcal{H} union the positive real axis. The quotient space \mathcal{P}/\sim_2 is homeomorphic to a truncated cylindrical surface, since the left and the right boundaries of \mathcal{P} are identified via \sim_2 for every fixed $\beta \in \mathcal{K}$. The third equivalence relation \sim_3 represents a particular case when the basis vectors have an identical length, i.e., $|b_1| = |b_2|$. Notice that \sim_3 is more involved compared to the other two equivalence relations, which renders a nontrivial geometry for \mathcal{L} and introduces complexities in defining a metric structure on \mathcal{L} .

To give more insights into the topologies of the lattice space \mathcal{L} , we present a couple of its subspaces formed by special types of lattices. In the first row of Figure 2.4, (a) is a square lattice $\Lambda\langle\beta, i\rangle$ with the red and blue arrows indicating two directions. The set of lattices obtained by stretching $\Lambda\langle\beta, i\rangle$ along these directions gives a subspace of \mathcal{L} displayed in

(b), which is homeomorphic to \mathbb{R} shown in (c). The origin of \mathbb{R} represents the square lattice $\Lambda\langle\beta, i\rangle$. A point on the positive side (the red half of (c)), r corresponds to a lattice of the form $\Lambda\langle\beta, i + r\rangle$; and a point on the negative side (the blue half of (c)), r corresponds to a lattice of the form $\Lambda\langle\beta e^{i\pi/2}, i - r\rangle$. Every lattice in this subspace is rectangular. In the second row of Figure 2.4, (d) shows a hexagonal lattice $\Lambda\langle\beta, e^{i\pi/3}\rangle$. In this case, stretching the hexagonal lattice along the three directions colored red, green, and blue gives a subspace of \mathcal{L} in (e), which is homeomorphic to a trifurcated structure shown in (f). It consists of three lines emerging from a common point that represents the hexagonal lattice $\Lambda\langle\beta, e^{i\pi/3}\rangle$; and each point on the lines corresponds to a unique non-hexagonal lattice.

On \mathcal{L} , we now construct a metric structure. As described above, there are three equivalence relations in \mathcal{L} . Given any two descriptor pairs $(\beta, \rho), (\beta', \rho') \in \mathcal{K} \times \mathcal{P}$, we define

$$D((\beta, \rho), (\beta', \rho')) = \sqrt{d_{\mathcal{K}}(\beta, \beta')^2 + d_{\mathcal{P}}(\rho, \rho')^2}, \quad (2.8)$$

where the equivalence relations \sim_1 and \sim_2 are incorporated into the definition of $d_{\mathcal{K}}$ and $d_{\mathcal{P}}$, respectively. Let $D_{\mathcal{K}}$ be a simple metric on \mathcal{K} , which separates the length differences and the angle differences as:

$$D_{\mathcal{K}}(\beta, \beta') = \sqrt{w(|\beta| - |\beta'|)^2 + (1 - w)(\cos^{-1} \frac{\operatorname{Re}(\beta \bar{\beta}')}{|\beta||\beta'|})^2}. \quad (2.9)$$

Here $w > 0$ is a parameter which adjusts the sensitivity between angle and length. We use $w = 0.05$ throughout this paper. The quotient metric on \mathcal{K} is then defined as

$$d_{\mathcal{K}}(\beta, \beta') = \min\{D_{\mathcal{K}}(\beta, \beta'), D_{\mathcal{K}}(-\beta, \beta')\}.$$

Let $D_{\mathcal{P}}$ be the well-known Poincaré metric [123] restricted to \mathcal{P} computed via

$$D_{\mathcal{P}}(\rho, \rho') = 2 \ln \frac{|\rho - \rho'| + |\rho - \bar{\rho}'|}{2\sqrt{\operatorname{Im}(\rho)\operatorname{Im}(\rho')}} ,$$

then the quotient metric for \mathcal{P}/\sim_2 is defined as

$$d_{\mathcal{P}}(\rho, \rho') = \min\{D_{\mathcal{P}}(\rho, \rho'), D_{\mathcal{P}}(\rho - 1, \rho'), D_{\mathcal{P}}(\rho + 1, \rho')\} .$$

Although $\rho \pm 1$ may fall outside of \mathcal{P} , the formula for $d_{\mathcal{P}}$ remains computationally valid, since the Poincaré metric is well defined everywhere in the upper half complex plane.

With D taking care of two equivalence relations, we consider the third one \sim_3 . It involves the lattices characterized by $|b_1| = |b_2|$ for the associated positive minimal basis (b_1, b_2) . When comparing lattices determined by (β, ρ) and (β', ρ') , there are 4 related points on the bottom arc of \mathcal{P} : $(\beta, e^{i\phi})$, $(e^{i\phi}\beta, -e^{-i\phi})$, $(\beta', e^{i\phi'})$ and $(e^{i\phi'}\beta', -e^{-i\phi'})$ for some $\phi, \phi' \in [\pi/3, 2\pi/3]$. The first pair of points, $(\beta, e^{i\phi})$ and $(e^{i\phi}\beta, -e^{-i\phi})$, represent a same lattice obtained by transforming $\Lambda(\beta, \rho)$; from the associated positive minimal basis (b_1, b_2) , $b_1 = \beta$ and $b_2 = \beta\rho$, keep b_1 unchanged, shrink the length of b_2 to be $|b_1| = |b_2|$, then rotate b_2 until its angle to b_1 is ϕ . Similarly, the second pair, $(\beta', e^{i\phi'})$ and $(e^{i\phi'}\beta', -e^{-i\phi'})$, is related to (β', ρ') . Moving between equivalent lattices does not induce length, thus a path connecting (β, ρ) and (β', ρ') while passing through one of these related points may be shorter than a direct path. In total, there are 8 types of such paths:

$$\begin{aligned} D_1 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \dashrightarrow (e^{i\phi}\beta, -e^{-i\phi}) \rightarrow (\beta', \rho') , \\ D_2 : (\beta, \rho) &\rightarrow (e^{i\phi'}\beta', -e^{-i\phi'}) \dashrightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') , \\ D_3 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \dashrightarrow (e^{i\phi}\beta, -e^{-i\phi}) \rightarrow (e^{i\phi'}\beta', -e^{-i\phi'}) \dashrightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') , \\ D_4 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \rightarrow (\beta', \rho') , \\ D_5 : (\beta, \rho) &\rightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') , \\ D_6 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \rightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') , \\ D_7 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \dashrightarrow (e^{i\phi}\beta, -e^{-i\phi}) \rightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') , \\ D_8 : (\beta, \rho) &\rightarrow (\beta, e^{i\phi}) \rightarrow (e^{i\phi'}\beta', -e^{-i\phi'}) \dashrightarrow (\beta', e^{i\phi'}) \rightarrow (\beta', \rho') . \end{aligned}$$

(2.10)

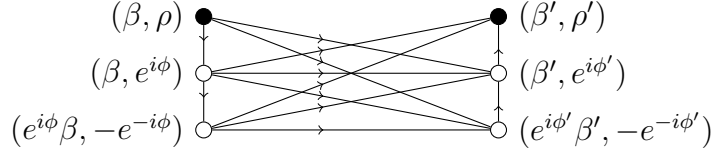


Figure 2.5: An illustration of the 8 types of paths, D_1 – D_8 in Equation 2.10 connecting (β, ρ) and (β', ρ') via 4 extra points in $\{(\beta_0, \rho_0) \mid \beta_0 \in \mathcal{K}, |\rho_0| = 1, \rho_0 \in \mathcal{P}\}$.

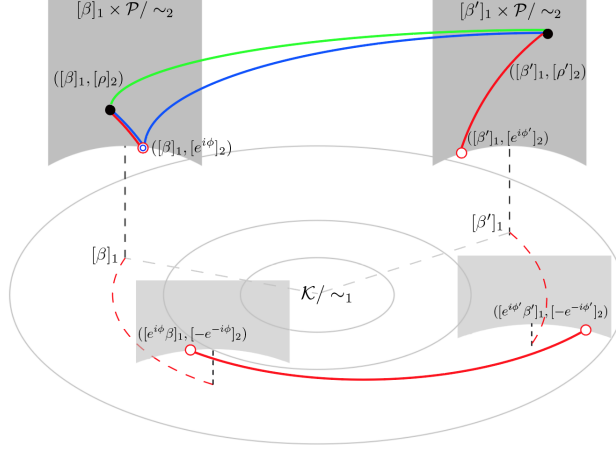


Figure 2.6: The lattice space \mathcal{L} is a product space $\mathcal{K}/\sim_1 \times \mathcal{P}/\sim_2$ modulo \sim_3 . The distance $d_{\mathcal{L}}((\beta, \rho), (\beta', \rho'))$ is the minimal length of the paths connecting (β, ρ) and (β', ρ') when the distance between equivalent points is reduced to 0. Here the green line shows D in Equation 2.8, the red line is D_3 in Equation 2.10, and the blue line is D_4 in Equation 2.10. Since D satisfies the triangle inequality, D is shorter than D_4 .

Here the pairs of lattices connected by \dashrightarrow are equivalent via \sim_3 , thus \dashrightarrow has 0 length; and the paths denoted by \rightarrow have lengths measured by D . Figure 2.5 illustrates these paths combinatorially showing that the list in Equation 2.10 is complete. Figure 2.6 exemplifies some paths in the lattice space \mathcal{L} : D in green, D_3 from Equation 2.10 in red, and D_4 from Equation 2.10 in blue.

We define the lattice metric as the length of the shortest path connecting (β, ρ) and (β', ρ') :

$$d_{\mathcal{L}}((\beta, \rho), (\beta', \rho')) = \min\{D, \min_{\phi, \phi' \in [\pi/3, 2\pi/3]} D_j(\phi, \phi'), j = 1, 2, 3\}, \quad (2.11)$$

where D is from Equation 2.8, and $\{D_j(\phi, \phi')\}_{j=1}^3$ are from Equation 2.10. Notice that it is not necessary to consider the cases with two consecutive \rightarrow in Equation 2.10. Since D satisfies the triangle inequality, a direct path is always shorter. For completeness, we present a pseudo-code for computing $d_{\mathcal{L}}$ in section A.1. Although $d_{\mathcal{L}}$ thus defined is a pseudometric, since $d_{\mathcal{L}}((\beta, \rho), (\beta', \rho')) = 0$ if and only if $[\beta, \rho] = [\beta', \rho']$, $d_{\mathcal{L}}$ is in fact a metric on \mathcal{L} .

Remark 2.3.1. *Notice that $d_{\mathcal{L}}$ is invariant under translation. It takes inputs from the lattice space \mathcal{L} , where only translational lattices are concerned. We may regard the visual difference between a lattice and its translated copy as a consequence of the boundedness of the image domain; thus, it is not intrinsic to the patterns.*

2.3.2 Sub-lattices and Parent-lattices in the Lattice Space

In section 2.2, we regard the collection of Möbius transforms as a group. Exploiting its subgroup, the modular group Γ allows us to address the problems of the basis representation. More generally, the group of Möbius transforms has a monoid structure where the inverse elements are not required compared with the definition of a group. Here we explore further the value of Möbius transforms by investigating one of its submonoids, $M_2(\mathbb{Z})$. We present the close relation between sub-lattices of a lattice and the monoid $M_2(\mathbb{Z})$. A one-to-one correspondence between sub-lattices and parent-lattices of a lattice is proved to extend this relation to that between parent-lattices and $M_2(\mathbb{Z})$. Such exploration also has practical significance. In section 2.5, when evaluating the lattice candidates, the confusion caused by moiré effects is eliminated by adding the density restriction in Equation 2.21. Figure 2.20 illustrates the necessity of this term numerically; here, we show the complexities from a theoretical perspective. The notions of sub- and parent-lattices can be algebraically defined using descriptors β and ρ .

Definition 2.3.2 (Sub-lattice). *Let $\Lambda = \Lambda\langle\beta, \rho\rangle$ and $\Lambda' = \Lambda\langle\beta', \rho'\rangle$ be two lattices. We say that Λ' is a sub-lattice of Λ , if there exists $\mathbf{k} = (k_1, k_2, k_3, k_4) \in \mathbb{Z}^4$ with $k_1k_4 - k_2k_3 > 0$*

such that

$$\begin{cases} \beta' = \beta(k_1 + k_2\rho) \\ \rho' = (k_3 + k_4\rho)/(k_1 + k_2\rho) \end{cases}.$$

$\Lambda\langle\beta', \rho'\rangle$ is said to be a sub-lattice of $\Lambda\langle\beta, \rho\rangle$ induced by \mathbf{k} .

This definition is derived from the equivalent expression:

$$\begin{cases} \beta' = k_1\beta + k_2\beta\rho \\ \beta'\rho' = k_3\beta + k_4\beta\rho \end{cases}, \quad (2.12)$$

with $k_1k_4 - k_2k_3 > 0$, which says that the basis for a sub-lattice comes from a non-degenerate linear combination of the basis of the original lattice using integer coefficients.

The set of transformations:

$$z \mapsto \frac{k_3 + k_4z}{k_1 + k_2z}, \{k_i\}_{i=1}^4 \subset \mathbb{Z}, \text{ such that } k_1k_4 - k_2k_3 > 0, \text{ for any } z \in \mathcal{H},$$

forms a monoid with function composition, which is denoted by $M_2(\mathbb{Z})$. In the category of monoids, $\text{PSL}_2(\mathbb{Z}) \leq M_2(\mathbb{Z}) \leq \text{PGL}_2(\mathbb{Z})$; hence, the discussion here is a generalization of section section 2.2. Symmetrically, we define parent-lattices as follows:

Definition 2.3.3 (Parent-lattice). *Let $\Lambda = \Lambda\langle\beta, \rho\rangle$ and $\Lambda' = \Lambda\langle\beta', \rho'\rangle$ be two lattices. We say that Λ' is a parent-lattice of Λ , if there exists $\mathbf{k} = (k_1, k_2, k_3, k_4) \in \mathbb{Z}^4$ with $v = 1/(k_1k_4 - k_2k_3) > 0$ such that*

$$\begin{cases} \beta' = v\beta(k_1 + k_2\rho) \\ \rho' = (k_3 + k_4\rho)/(k_1 + k_2\rho) \end{cases}.$$

Λ' is said to be a parent-lattice of Λ induced by \mathbf{k} .

This definition says that for a parent-lattice (β', ρ') of (β, ρ) , there exist constants a, b, c, d with $u = ad - bc > 0$ such that

$$\begin{cases} \beta' = a\beta + b\rho \\ \beta'\rho' = c\beta + d\rho \end{cases} \iff \begin{cases} \beta = d\beta'/u - b\beta'\rho'/u \\ \beta\rho = -c\beta'/u + a\beta'\rho'/u \end{cases}. \quad (2.13)$$

Note that only when a, b, c, d are integers that Λ' becomes a sub-lattice of Λ . Comparing the left side of Equation 2.13 with Definition 2.3.3, we observe that $a = vk_1$, $b = vk_2$, $c = vk_3$ and $d = vk_4$, and $u = v^2(k_1k_4 - k_2k_3) = v$, thus the right side of Equation 2.13 becomes:

$$\begin{cases} \beta = \beta'(k_4 - k_2\rho') \\ \rho = (-k_3 + k_1\rho')/(k_4 - k_2\rho') \end{cases}. \quad (2.14)$$

From Definition 2.3.2, we see that Λ' is a parent-lattice of Λ , if Λ is a sub-lattice of Λ' induced by $(k_4, -k_2, -k_3, k_1)$. By checking the equivalence relations among descriptors, we have the following proposition:

Proposition 2.3.1. *Given a lattice $\Lambda = \Lambda\langle\beta, \rho\rangle$, there is a one-to-one correspondence:*

$$\{\text{Sub-lattices of } \Lambda\} \xleftrightarrow{\varphi} \{\text{Parent-lattices of } \Lambda\} \quad (2.15)$$

well-defined as follows: if \mathbf{k} determines a sub-lattice via Definition 2.3.2, then it determines a parent-lattice via Definition 2.3.3.

Figure 2.7 shows an example of parent- and sub-lattices. Once all the sub-lattices of a lattice are found, a complete set of its parent-lattices comes for free by employing Equation 2.15.

Finding all the sub-lattices of a lattice with shape descriptor $\rho \in \mathcal{P}$ is equivalent to searching for all the elements in $M_2(\mathbb{Z})$ that send ρ back to \mathcal{P} . It suffices to see its action

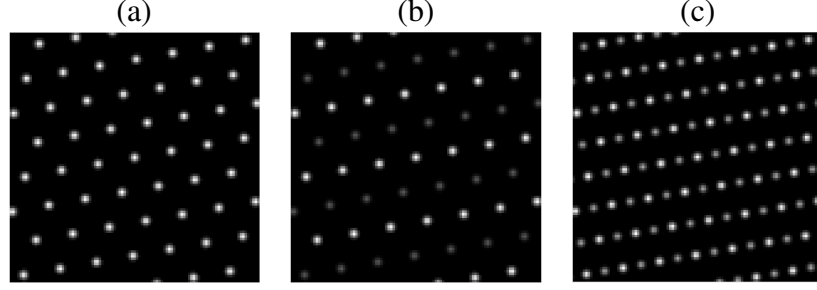


Figure 2.7: One-to-one correspondence between the sub- and the parent-lattices. (a) A lattice $\Lambda\langle\beta, \rho\rangle = \Lambda\langle 14.7721 + 2.6047i, e^{i\pi/3}\rangle$. (b) A sub-lattice $\Lambda\langle\beta, 2\rho + 1\rangle$ in white, with lattice (a) in gray. (c) A parent-lattice $\Lambda\langle\beta/2, 2\rho + 1\rangle$ of (a). The common particles are emphasized with white color.

on three distinct points by the property of the Möbius transformation. Generally, given a $\rho \in \mathcal{P}$, finding all such integer coefficients is difficult and not fruitful. There are two families of sub-lattices, which are easy to consider:

1. For $\Lambda = \Lambda(\beta, \rho) \in \mathcal{L}$ with $|\rho| \geq n$, for an arbitrary integer $n \geq 1$, $\Lambda(m\beta, \rho/m)$ is a sub-lattice of Λ induced by $\mathbf{k} = (m, 0, 0, 1)$ for any $m \leq n, m \in \mathbb{N}$; the corresponding $M_2(\mathbb{Z})$ -action is $z \mapsto z/m$ for $z \in \mathbb{C}$.
2. For $\Lambda = \Lambda(\beta, \rho) \in \mathcal{L}$ with $|\operatorname{Re}(\rho)| \leq 1/(2n)$, for an arbitrary integer $n \geq 1$, $\Lambda(\beta, m\rho)$ is a sub-lattice of Λ induced by $\mathbf{k} = (1, 0, 0, m)$ for any $m \leq n, m \in \mathbb{N}$; the corresponding $M_2(\mathbb{Z})$ -action is $z \mapsto mz$ for $z \in \mathbb{C}$.

In some cases, it can be easy to find conditions for $\mathbf{k} \in \mathbb{Z}^4$ whose associated action sends $\rho \in \mathcal{P}$ to \mathcal{P} . For example, when $k_2 = 0$ (which forces $k_1 \neq 0$), we find $\infty \rightarrow \infty$, $0 \mapsto k_3/k_1$ and $\pm 1/2 \mapsto (\pm k_4/2 + k_3)/k_1$ by the $M_2(\mathbb{Z})$ -action determined by this (k_1, k_2, k_3, k_4) . In order to have a non-empty intersection with \mathcal{P} , we must require:

$$\min\{(\pm k_4/2 + k_3)/k_1\} \leq 1/2 \text{ or } k_3^2 - k_4^2/4 < 0 \text{ and } \min\{(\pm k_4/2 + k_3)/k_1\} \geq 1/2 .$$

By Proposition 2.3.1, these results also extend symmetrically to parent-lattices.

2.4 Validation of the Lattice Space \mathcal{L} and Metric $d_{\mathcal{L}}$

2.4.1 Visual Validation

For the purpose of comparison, one may assign the following 4-tuple to a lattice with a positive minimal basis (b_1, b_2) :

$$(|b_1|, |b_2|, \theta, \psi) = (|b_1|, |b_2|, \text{Arg } b_1, \cos^{-1}(\frac{\text{Re}(b_1 \bar{b}_2)}{|b_1||b_2|})) . \quad (2.16)$$

Here $\theta \in (-\pi/2, \pi/2]$ is the angle from the positive real axis to b_1 , and $\psi \in [\pi/3, 2\pi/3]$ is the angle between b_1 and b_2 . Similar to $d_{\mathcal{L}}$, the differences in these parameters indicate visual differences between lattice patterns; however, $d_{\mathcal{L}}$ is more stable and consistent. Figure 2.8 compares the 4-tuple representations in Equation 2.16 and $d_{\mathcal{L}}$. The pair of very similar lattices, $\Lambda_A = \Lambda(12, 12.5, 10^\circ, 90^\circ)$ in (a) and $\Lambda_B = \Lambda(12, 12.5, -80^\circ, 90^\circ)$ in (b) show a large relative difference in θ : 900%; while their metric distance $d_{\mathcal{L}} = 0.0816$ is short. In general, when $|b_1| \approx |b_2|$, minor numerical errors trigger large relative errors in θ -component due to the equivalence relations. The lattices (a) Λ_A , (c) $\Lambda_C = \Lambda(13, 13.5, 10^\circ, 85^\circ)$, and (d) $\Lambda_D = \Lambda(12.5, 13.5, 11^\circ, 91^\circ)$ are more distinguishable. Equation 2.16 shows the difference in multiple numbers; in contrast, $d_{\mathcal{L}}$, as a single value, provides compact information integrating various aspects of the visual differences. This feature of $d_{\mathcal{L}}$ allows a simple lattice pattern comparison.

Figure 2.9 (a)–(e) present five different lattice patterns and their pairwise distances in \mathcal{L} . Visually, lattice Λ_A is more different from Λ_C than from Λ_B , and the corresponding distances, $d_{\mathcal{L}}(\Lambda_A, \Lambda_C) = 0.7083 > d_{\mathcal{L}}(\Lambda_A, \Lambda_B) = 0.5493$, are consistent with this observation. Among all the pairs of the five lattices, Λ_A and Λ_D are the most similar ones, and accordingly, $d_{\mathcal{L}}(\Lambda_A, \Lambda_D) = 0.0203$ is the shortest distance. The difference between Λ_B and Λ_C , and that between Λ_D and Λ_E are similar, which is well represented by the distance $d_{\mathcal{L}}(\Lambda_B, \Lambda_C) = 0.4472$ being close to $d_{\mathcal{L}}(\Lambda_D, \Lambda_E) = 0.4472$. This is also the case for the

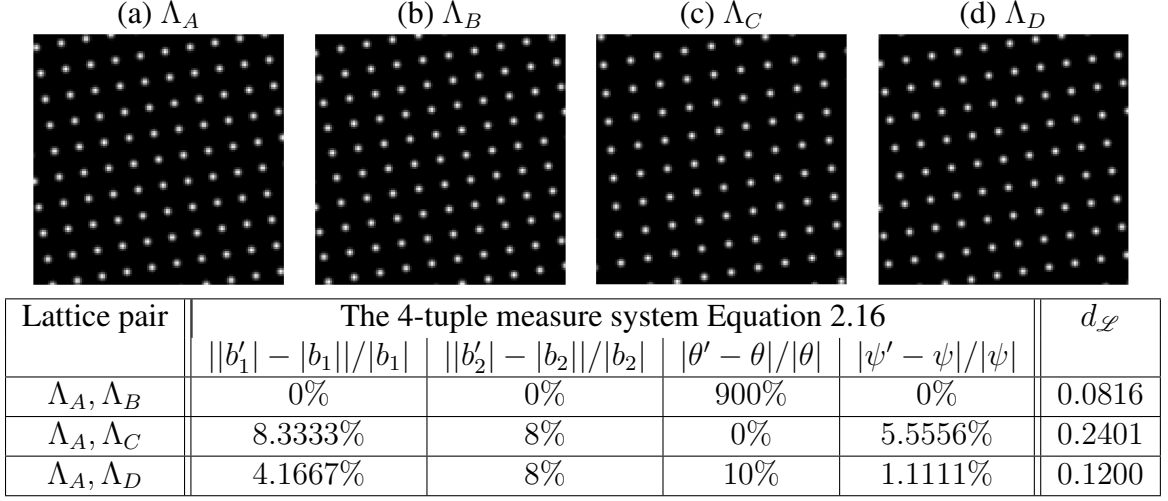


Figure 2.8: Metric comparison. Lattice (a) $\Lambda_A = \Lambda(11.8177 + 2.0838i, -2.1706 + 12.3101i)$ and (b) $\Lambda_B = \Lambda(2.0838 - 11.8177i, 12.3101 + 2.1706i)$ are visually similar. The 4-tuple measure indicates a significant difference in θ , while $d_{\mathcal{L}}$ gives a small value. The lattices (a), (c) $\Lambda_C = \Lambda(-1.1766 + 13.4486i, -2.0838 + 11.8177i)$ and (d) $\Lambda_D = \Lambda(11.8177 + 2.0838i, -2.1706 + 12.3101i)$ are more distinguishable, but the differences are scattered in four numbers using (Equation 2.16). $d_{\mathcal{L}}$ integrates these differences and provides a compact measure.

pair Λ_B and Λ_D , compared to the pair Λ_C and Λ_E , where $d_{\mathcal{L}}(\Lambda_B, \Lambda_D) = 0.5293$ is close to $d_{\mathcal{L}}(\Lambda_C, \Lambda_E) = 0.5293$.

2.4.2 Quantitative Validation

The representation $\Lambda\langle\beta, \rho\rangle$ provides a universal framework to characterize lattices. For example, $\Lambda\langle 10, i\rangle$ represents a cubic lattice; $\Lambda\langle 10, e^{i\pi/3}\rangle$ denotes a hexagonal lattice; and the notation $\Lambda\langle e^{-2\pi i/9}, e^{4\pi i/9}\rangle$ represents a centered rectangular lattice.

A useful feature of $d_{\mathcal{L}}$ is that it can adjust sensitivity to orientation versus scale. Depending on the application, we can change the weight parameter w in Equation 2.9 to emphasize the inconsistency in angles or lengths. When $w = 0$, only the lattice orientation is considered, which is similar to [104]. Figure 2.10 (a) shows $d_{\mathcal{L}}(\Lambda\langle 10, e^{i\pi/3}\rangle, \Lambda\langle (10 + \Delta|\beta|)e^{i\pi/9}, e^{i\pi/3}\rangle)$ when $w = 0.5, 0.05$ and 0.005 , where the scale variation $\Delta|\beta| \in [-5, 5]$ and orientation difference is fixed at $e^{i\pi/9}$.

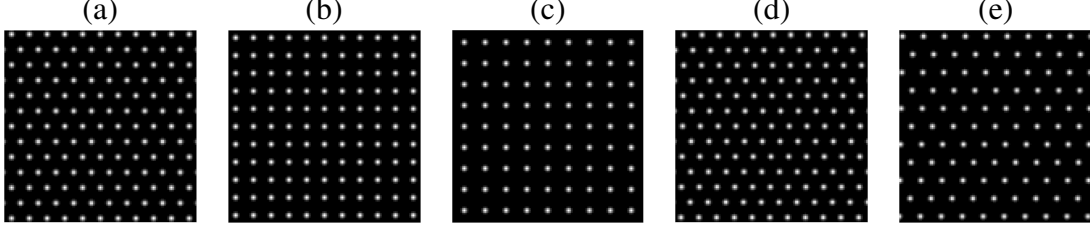


Figure 2.9: Visual effects of $d_{\mathcal{L}}$. Five different lattices: (a) $\Lambda_A = \Lambda\langle 11, e^{i\pi/3} \rangle$, (b) $\Lambda_B = \Lambda\langle 11, e^{i\pi/2} \rangle$, (c) $\Lambda_C = \Lambda\langle 13, e^{i\pi/2} \rangle$, (d) $\Lambda_D = \Lambda\langle 11, e^{i61\pi/180} \rangle$, and (e) $\Lambda_E = \Lambda\langle 13, e^{i61\pi/180} \rangle$ are displayed. Pairwise distances: $d_{\mathcal{L}}(\Lambda_A, \Lambda_B) = 0.5493$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_C) = 0.7083$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_D) = 0.0203$, $d_{\mathcal{L}}(\Lambda_A, \Lambda_E) = 0.4477$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_C) = 0.4472$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_D) = 0.5293$, $d_{\mathcal{L}}(\Lambda_B, \Lambda_E) = 0.6929$, $d_{\mathcal{L}}(\Lambda_C, \Lambda_D) = 0.6929$, $d_{\mathcal{L}}(\Lambda_C, \Lambda_E) = 0.5293$, and $d_{\mathcal{L}}(\Lambda_D, \Lambda_E) = 0.4472$ are computed. They are consistent with the visual perception of the lattice differences.

Typical configuration of atoms present hexagonal patterns, and to $d_{\mathcal{L}}$, the misorientation between hexagonal lattices is more distinguishable. Figure 2.10 (b) shows $d_{\mathcal{L}}(\Lambda\langle 10, e^{\pi i/3} \rangle, \Lambda\langle \beta, \rho \rangle)$ for a set of different β with unit ρ , whose arguments varies within $[\pi/3, 2\pi/3]$. Notice that $d_{\mathcal{L}}$ has the biggest differences when $\Delta \text{Arg } \rho = 0$ and $\Delta \text{Arg } \rho = 60^\circ$ corresponding to the left and right boundary respectively, which produces hexagonal lattices. This shows that $d_{\mathcal{L}}$ is most sensitive to the misorientation between hexagonal lattices. Since rotating a hexagonal by 30° clockwise and counter-clockwise result in an identical lattice, we see the red curve ($\text{Arg } \beta = 30^\circ$) is mirror-symmetric. The two global minima of this red curve bring up an important property that the Riemannian center in $(\mathcal{L}, d_{\mathcal{L}})$ is not unique. As a consequence, we can not directly apply centroid-based analysis.

From a different perspective, in Figure 2.10 (c), we compute $d_{\mathcal{L}}(\Lambda\langle 10, \rho \rangle, \Lambda\langle \beta, \rho \rangle)$ for a set of different ρ with β satisfying $|\beta| = 10$ and $\text{Arg } \beta$ varies within $[0, \pi]$. Since $\Lambda\langle 10, e^{\pi i/3} \rangle$, $\Lambda\langle 10e^{\pi i/3}, e^{\pi i/3} \rangle$ and $\Lambda\langle 10e^{2\pi i/3}, e^{\pi i/3} \rangle$ are equivalent (hexagonal) lattices, and $\Lambda\langle 10, e^{\pi i/2} \rangle$ and $\Lambda\langle 10e^{\pi i/2}, e^{\pi i/2} \rangle$ are equivalent (rectangular) lattices, their distances are 0 respectively. Figure 2.10(b) and (c) also show that lattice metric $d_{\mathcal{L}}$ is able to measure the differences between lattices of any Bravais lattice types. The comparison using $d_{\mathcal{L}}$ considers differences between equivalence classes of lattice representations, which is

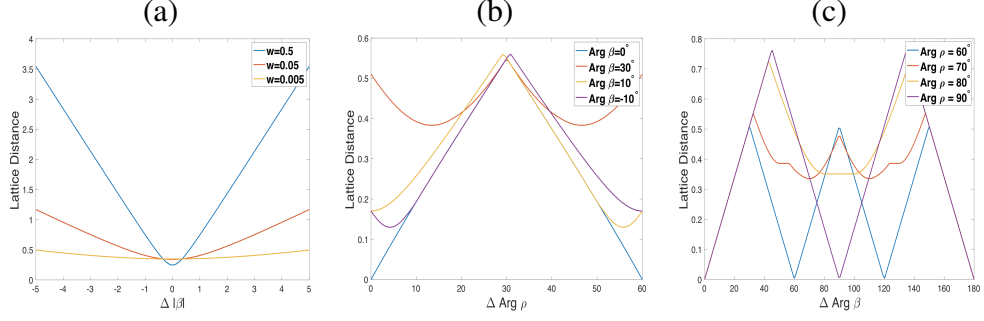


Figure 2.10: Properties of $d_{\mathcal{L}}$. (a) Effect of changing w . (b) Misorientation of hexagonal lattices is emphasized using $d_{\mathcal{L}}$, which corresponds to the left and right edges. (c) High symmetry of the hexagonal lattice is reflected by the symmetry of the blue curve. Lattices to be compared are not necessarily of the same type, and $d_{\mathcal{L}}$ considers the lattice equivalence relations.

different from [104] where lattice differences are measured via intensities between image patches.

2.5 Application to Error Quantification of Lattice Identification and Separation Algorithm (LISA)

In this section, we describe the Lattice Identification and Separation Algorithm (LISA), which is used to separate superposed lattices: a mixture of multiple two-dimensional lattices laid over another. This structure is referred to as a **superlattice** [137]. Superlattices are explored in solid physics [138, 139, 140], surface waves [141, 142] and nonlinear optics [143]. One of the most significant discoveries in low-dimensional material sciences is the family of transition metal dichalcogenides (TMDs) [144, 145], such as MoS_2 [146] and WTe_2 [147]. A single sheet of TMD shows a superlattice structure: the top and the bottom are lattice layers of chalcogen atoms, and the middle is a lattice layer of transition metal atoms. The main idea behind LISA is to measure the periodicities globally by the Fourier transform. For higher accuracy of the lattice basis estimation, we exploit the Fourier Slice Theorem [114]. By evaluating pairs of peaks on the power spectrum, we find the optimal lattice structure. Also, we use a stepwise refinement to obtain a stable estimation. The pro-

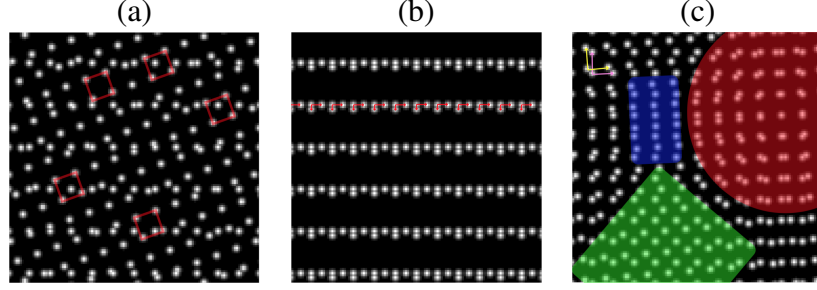


Figure 2.11: Challenges of pattern separation. Each image above has two lattices superposed. (a) The red boxes indicate textons of a single lattice, and they have different interiors which can confuse the texton-based methods. (b) Using non-superposed lattice identification methods, wrong local features (e.g., L-shapes [115], shown as the red arrows) can be identified. These red arrows do not correspond to any of the true underlying lattices. (c) The pink and the yellow L-shapes in the upper-left corner denote the true lattice components. The moiré patterns indicated by the red, blue, and green regions are different from the underlying lattices.

posed method is designed to handle moiré effects, excessive density, and inhomogeneous texton interiors.

We analytically study the properties of LISA. In particular, we explore the effects of particle radius, lattice density, and relative translations on LISA’s performances, and show that LISA is robust against the Gaussian perturbation with bounded variation.

Separating individual lattice patterns from a superlattice is challenging. First, it is difficult to determine the unit, e.g., the texton [148] and the L-shape [115]. Effective methods for non-superposed lattices, such as [149, 115] may fail due to the interaction from different lattice layers. Figure 2.11 (a) shows a typical situation where the textons of one lattice have inhomogeneous interiors, and (b) shows where local L-shapes do not represent the correct underlying patterns.

Secondly, superposed periodic patterns may produce new periodic structures, i.e., the moiré patterns [150], which can confuse the identification process. (This phenomenon is exploited in applications such as [151, 152].) Figure 2.11 (c) shows three different moiré patterns generated by two lattices, whose bases are represented by the pink and yellow L-shapes in the upper-left corner. Thirdly, human supervision [153] can be unreliable.

Psychological evidence [154, 148, 155, 156] proves that similarities in geometry can hinder the visual search. For example, less than 15° of rotational differences between the targets and the background increase errors [155]; and small variations in densities can interfere with the target identification [148].

2.5.1 Variational Model for Lattice Separation

For a given image with a mixture of lattices $U : \Omega \subseteq \mathbb{R}^2 \rightarrow [0, 1]$ as defined in Equation 2.1, assuming that σ is sufficiently small, we propose to identify the underlying lattice patterns by minimizing the following energy:

$$\min_{N \in \mathbb{N}, \Lambda_j \in \mathcal{L}, \mu_j \in \mathbb{C}} \int_{\Omega} |U - \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j| dx dy + hN, \quad (2.17)$$

where $dx dy$ is the Lebesgue measure on \mathbb{R}^2 , and $h > 0$ is a penalty coefficient. The regularization term hN helps to avoid identifying multiple sub-lattices from a single dense lattice. It suppresses the number of lattice layers when fitting a mixture to the given image.

For a fixed N , this energy is balancing two competing terms. Using $|a - b| = a + b - 2 \min(a, b)$ for any $a, b \in \mathbb{R}$, the minimization of Equation 2.17 becomes

$$\min_{\substack{\Lambda_j \in \mathcal{L} \\ \mu_j \in \mathbb{C}}} \left\{ \int_{\Omega} U - \min(U, \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j) dx dy + \int_{\Omega} \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j - \min(U, \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j) dx dy \right\}. \quad (2.18)$$

These integrals are equivalent to counting particles. Let U_p denote the set of locations of particles in U , and then we have the following correspondences:

$$\begin{aligned} \int_{\Omega} U - \min(U, \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j) dx dy &\iff U_p - U_p \bigcap \bigcup_{j=1}^N (\Lambda_j + \mu_j), \\ \int_{\Omega} \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j - \min(U, \max_{j=1, \dots, N} \mathcal{T}_{\mu_j} \Lambda_j) dx dy &\iff \bigcup_{j=1}^N (\Lambda_j + \mu_j) - U_p \bigcap \bigcup_{j=1}^N (\Lambda_j + \mu_j). \end{aligned}$$

The sets on the right sides can be further expressed as

$$\bigcap_{j=1}^N (U_p \cap (\Lambda_j + \mu_j)^c) , \text{ and } \bigcup_{j=1}^N ((\Lambda_j + \mu_j) \cap U_p^c) ,$$

respectively. The problem in Equation 2.18 is thus equivalent to

$$\min_{\substack{\Lambda_j \in \mathcal{L} \\ \mu_j \in \mathbb{C}}} \left\{ \underbrace{\int_{\Omega} U - \max_{j=1, \dots, N} (\min(U, \mathcal{T}_{\mu_j} \Lambda_j)) dx dy}_{\text{under-fitting}} + \underbrace{\int_{\Omega} \max_{j=1, \dots, N} (\mathcal{T}_{\mu_j} \Lambda_j - \min(U, \mathcal{T}_{\mu_j} \Lambda_j)) dx dy}_{\text{over-fitting}} \right\} . \quad (2.19)$$

The first term in the objective function measures the remaining intensities of U after points are extracted by N lattices, i.e., the under-fitting. The second term evaluates the total excessive intensities of these N lattices, i.e., the over-fitting. As N increases, the under-fitting decreases. If we control the over-fitting to be 0, i.e., each lattice candidate has no extra lattice points, then by including more layers, Equation 2.19 reaches the minimum. Therefore, we solve Equation 2.19 using a greedy strategy, which leads to LISA in the following section.

2.5.2 Lattice Identification and Separation Algorithm (LISA)

We present the outline of the algorithm in algorithm 1, and a demonstration of its workflow is in Figure 2.12.

In real applications, the size, the shape, and the intensity of each particle may differ, which complicates the identification. After background denoising if necessary (e.g., using the Otsu's method [157]), we process the image by replacing each local maximum on the intensity surface with a common Gaussian PSF G_{σ} . We denote this processing by \mathcal{F} :

$$\mathcal{F}(U) = G_{\sigma} * \delta(|\nabla U|) . \quad (2.20)$$

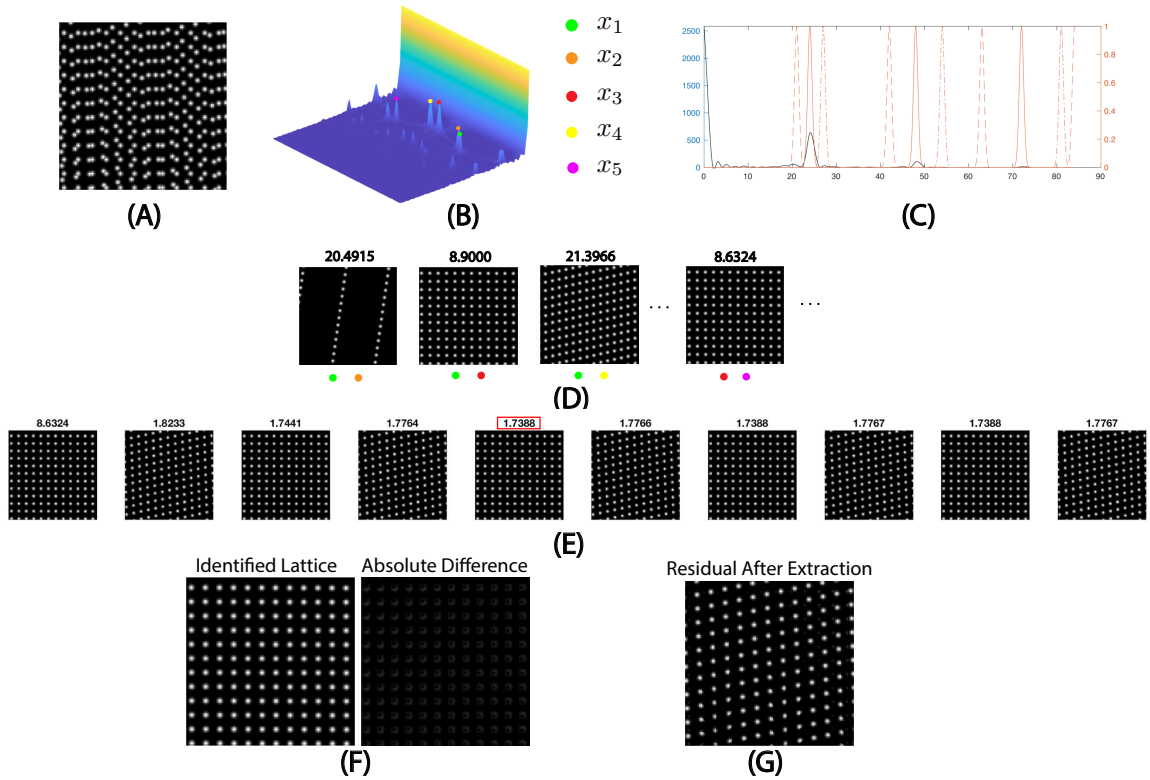


Figure 2.12: Steps of LISA. (A) An image processed by (Equation 2.20). **Step 1:** (B) The power spectrum on the polar coordinate, and the high responses using $J = 5$. (C) Peak locations refined via matching Gaussian impulses. **Step 2:** (D) Generate lattice candidates $\mathcal{T}_{\mu_{k,l}}\Lambda_{(k,l)}$, $k, l = 1, \dots, 5, k \neq l$, for each pair of high peaks, and compute their energies (Equation 2.21). Pick (x_3, x_5) (red and purple in (B)) to be the optimal $\mathcal{T}_{\mu_1}\Lambda_1$, since it has the lowest energy. **Step 3:** (Optional) (E) Update $\mathcal{T}_{\mu_1}\Lambda_1$ with $\mathcal{T}_1\Lambda_{\mu_1}^{(5)}$. **Step 4:** (F) The optimal lattice $\mathcal{T}_{\mu_1}\Lambda_1$ identified in this iteration; and the absolute difference between $\mathcal{T}_{\mu_1}\Lambda_1$ and the underlying true lattice. The absolute difference has an average value of 0.0202, and maximum of 0.1924, showing the effectiveness of LISA. (G) The remainder image. The average intensity 0.0710 is greater than the accuracy criterion 0.01; thus, proceed to the next iteration.

Inputs:

1. U : the given gray scale image of a superlattice;
2. J : a parameter to control the number of lattice candidates;
3. (Optional) K : the number of iterations for the refinement.

Let $j = 1$. While TRUE:

Step 1. Compute the Fourier transform of U on the polar coordinate. Collect the local maxima on the power spectrum, $C_j = \{x_1, x_2, \dots, x_M\}$ within J connected components.

Step 2. For every pair $(x_k, x_l) \in C_j$, $k \neq l$, construct a lattice pattern and compute the translation $\mu_{k,l}$ to get $\mathcal{T}_{\mu_{k,l}}\Lambda_{(k,l)}$. Take $\mathcal{T}_{\mu_j}\Lambda_j = \arg \min_{k,l=1,\dots,M;k \neq l} \mathcal{E}(\mathcal{T}_{\mu_{k,l}}\Lambda_{(k,l)})$ as in (Equation 2.21).

Step 3. (Optional) K -step-refinement of $\mathcal{T}_{\mu_j}\Lambda_j$.

Step 4. For the identified optimal candidate $\mathcal{T}_{\mu_j}\Lambda_j$, if $\text{mean}(U - \mathcal{T}_{\mu_j}\Lambda_j) < 0.01$, terminate the algorithm; otherwise, update $U = \mathcal{F}(U - \mathcal{T}_{\mu_j}\Lambda_j)$, $j = j+1$ and repeat.

Algorithm 1: Lattice Identification and Separation Algorithm (LISA)

For images with low or medium resolution, we apply the Gaussian approximation method [158] to calibrate the peak locations (also see [159, 160] for other peak localization methods). Letting $U(x, y)$ be a discrete local maximum, i.e., $U(x, y) \geq U(x', y')$, for $x' = x \pm 1$ and $y' = y \pm 1$, the calibrated coordinate (\hat{x}, \hat{y}) for the peak (x, y) is computed via

$$\begin{aligned}\hat{x} &= x - \frac{\log(U(x+1, y)) - \log(U(x-1, y))}{2(\log(U(x+1, y)) + \log(U(x-1, y)) - 2\log(U(x, y)))}, \\ \hat{y} &= y - \frac{\log(U(x, y+1)) - \log(U(x, y-1))}{2(\log(U(x, y+1)) + \log(U(x, y-1)) - 2\log(U(x, y)))}.\end{aligned}$$

In Step 1, we compute the Radon transform of the image by a B-spline convolution-based method [161]. The result is a 1D signal for each projecting angle, upon which we apply the standard 1D FFT. The collection of these 1D spectra form the 2D Fourier transform of the image on the polar coordinate (see Theorem 2.1.1), see Figure 2.12 (B). For computational efficiency, we focus on peaks with sufficient heights. We set the threshold such that above which, the power spectrum has J connected components, e.g., $J = 5$ and

the filtered peaks x_1, \dots, x_5 in Figure 2.12 (B). To achieve sub-pixel precision, we adjust the peak locations by perturbing the period. Figure 2.12 (C) demonstrates this process: consider trains of Gaussian impulses placed periodically along the radial direction; by perturbing the period from the origin, we choose the one that overlaps with the signal the most and select its period to be the adjusted distance.

In Step 2, each pair of local maxima on the power spectrum corresponds to a lattice candidate in the image domain. Figure 2.12 (D) displays 4 examples of such combinations. The Fourier transform of a lattice $\Lambda(b_1, b_2)$ in the image domain is a lattice in the frequency domain, called its reciprocal lattice $\Lambda(\omega_1, \omega_2)$. This relation is given by:

$$\begin{cases} [b_1 \mid 0]^T = ([\omega_2 \mid 0]^T \times [0, 0, 1]^T) / (|[\omega_1 \mid 0]^T \times [\omega_2 \mid 0]^T \cdot [0, 0, 1]^T|) \\ [b_2 \mid 0]^T = ([0, 0, 1]^T \times [\omega_1 \mid 0]^T) / (|[\omega_1 \mid 0]^T \times [\omega_2 \mid 0]^T \cdot [0, 0, 1]^T|) \end{cases}.$$

We identify the translation for each candidate by finding the maximum of the cross-correlation between the candidate and the original image. To evaluate the lattice candidates shown in Figure 2.12 (D), we propose the following energy:

$$\mathcal{E}(\mathcal{T}_\mu \Lambda) = \underbrace{\|\mathcal{F}(U - \mathcal{T}_\mu \Lambda) \odot \mathcal{F}(U)\|_2}_{\text{under-fitting}} + \gamma \underbrace{\left| \frac{\#\mathcal{T}_\mu \Lambda}{\#\mathcal{F}(\mathcal{T}_\mu \Lambda \odot U) + \varepsilon} - 1 \right|}_{\text{over-fitting}}, \quad \Lambda \in \mathcal{L}, \mu \in \mathbb{C}. \quad (2.21)$$

Here U denotes the original image, $\# \cdot$ counts the number of particles, and \odot is the element-wise multiplication of matrices. We truncate the intensity differences between images so that negative values are replaced by 0. $\gamma > 0$ is a penalty coefficient (we set $\gamma = 10$), and $\varepsilon > 0$ is a small constant to avoid division by 0 (we set $\varepsilon = 1 \times 10^{-8}$).

Notice the similarities between Equation 2.21 and the target function in Equation 2.19. The first component in Equation 2.21 measures the portion of particles not covered by the lattice candidate, i.e., the under-fitting. A small value represents that more particles in the

image are identified with the lattice points of $\mathcal{T}_\mu\Lambda$. We normalize the remainder $U - \mathcal{T}_\mu\Lambda$ to be comparable with $\mathcal{F}(U)$. The element-wise multiplication with $\mathcal{F}(U)$ prevents new points generated by incomplete particles after the extraction. The second term in Equation 2.21 compares the ratio between the number of lattice points of the candidate and what is identified in the image, i.e., the over-fitting. Therefore, the optimal candidate lattice has minimal energy.

Step 3, illustrated in Figure 2.12 (E), is similar to a sampling procedure with replacement. This step is optional, yet when the number of underlying lattices is large, it improves the accuracy of identification. As shown in subsection 2.5.3, superposing lattices complicate the power spectrum; the early identification is affected the most, yielding unstable identifications for the remaining lattices. This optional step refines the results, and proceeds iteratively. Initialize $t = 1$ and let $\mathcal{T}_{\mu_j}\Lambda_j^{(1)} = \mathcal{T}_{\mu_j}\Lambda_j$. For $t = 1, \dots, K$, compute and normalize the remainder $\mathcal{F}(U - \mathcal{T}_{\mu_j}\Lambda_j^{(t)})$. Then iterate Step 1 and Step 2 on $\mathcal{F}(U - \mathcal{T}_{\mu_j}\Lambda_j^{(t)})$ to find the next candidate, $\mathcal{T}_{\mu_j}\Lambda_j^{(t+1)}$. After K such candidates are generated, as in Figure 2.12 (E), update $\mathcal{T}_{\mu_j}\Lambda_j$ with the one giving the minimal energy (the red in Figure 2.12 (E)). This is the output for this iteration of LISA, shown in Figure 2.12 (F).

In Step 4, the optimal candidate $\mathcal{T}_{\mu_j}\Lambda_j$ is subtracted from the original image, and the difference is truncated so that negative values are replaced by 0. Figure 2.12 (G) shows the remainder image. We compute the average intensity of the residual image $U - \mathcal{T}_{\mu_j}\Lambda_j$ and terminate the algorithm if it is smaller than the accuracy criterion 0.01; otherwise, we preprocess the residual using \mathcal{F} , update the original image, and repeat Step 1–4.

2.5.3 Analytical Properties of LISA: Superlattice and Spectrum Surface

We describe the close relation between LISA and the geometric features of a superlattice. Assuming that in (Equation 2.1), the remainder image has 0 intensity, i.e., $R = 0$, and $\max_{j=1, \dots, N} \mathcal{T}_{\mu_j}\Lambda(b_{j,1}, b_{j,2}) \approx \sum_{j=1, \dots, N} \mathcal{T}_{\mu_j}\Lambda(b_{j,1}, b_{j,2})$, the Fourier transform of a super-

lattice image becomes

$$\hat{U}(\xi) = \hat{G}_\sigma(\xi) \sum_{j=1}^N \frac{\Lambda_j^*(\xi)}{\det \Lambda_j} \exp(-i2\pi\xi \cdot \mu_j), \quad \xi \in \mathbb{R}^2. \quad (2.22)$$

Here, ξ represents the frequency coordinate; \hat{G}_σ corresponds to the Fourier transform of the PSF G_σ ; and the rotation $\exp(-i2\pi\xi \cdot \mu_j)$ is due to the shift μ in the image domain. The Fourier transform of the lattice impulse Λ_j , $j = 1, \dots, N$, consists of its reciprocal lattice impulse Λ_j^* modulated by $\det \Lambda_j$, the fundamental volume of $\Lambda_j = \Lambda\langle\beta_j, \rho_j\rangle$ computed via $\text{Im}(\overline{\beta_j}\beta_j\rho_j)$. The reciprocal lattice impulse Λ_j^* can be expressed in the lattice space by:

$$[\hat{\beta}_j, \hat{\rho}_j] = \left[\frac{\beta_j \exp(-i\pi/2)}{\det \Lambda_j}, \rho_j \right] \in \mathcal{L}.$$

Equation 2.22 implies that the Fourier transform of a superlattice image is a mixture of complex lattices influenced by three factors: the centered Gaussian \hat{G}_σ , the fundamental volumes $\det \Lambda_j$, and the lattice translations μ_j , $j = 1, 2, \dots, N$. Without these modifications, every lattice in the image domain corresponds to two peaks on the power spectrum. Notice that for $\xi \in \mathbb{R}^2$, $\Lambda_j^*(\xi) = 1$ if and only if $1/|\xi|$ is a period of Λ_j along the direction of ξ . Hence, we can identify lattices $\{\Lambda_j\}_{j=1}^N$ with correct combinations of the peaks on the power spectrum surface, and we apply this in Step 2 of LISA.

The Gaussian PSF and the fundamental volumes of lattices complicate the problem. First, independent of the positions of the superlattice particles, a centered Gaussian \hat{G}_σ globally dampens the power spectrum. If $|\xi|$ is small, $\hat{G}_\sigma(\xi)$ has little influence on the power spectrum, and if $|\xi|$ is large, $\hat{G}_\sigma(\xi)$ considerably decreases the power at ξ . Second, the radius of particles controls the rate of radial decay of the power spectrum surface. High-frequency components are preserved if the particles of the superlattice have a small radius, as the standard deviation σ is small. Third, the fundamental volumes of the original lattices affect the power spectrum. The magnitudes of a pair of peaks on the spectrum surface associated with the lattices with the smaller fundamental volumes are augmented, and those

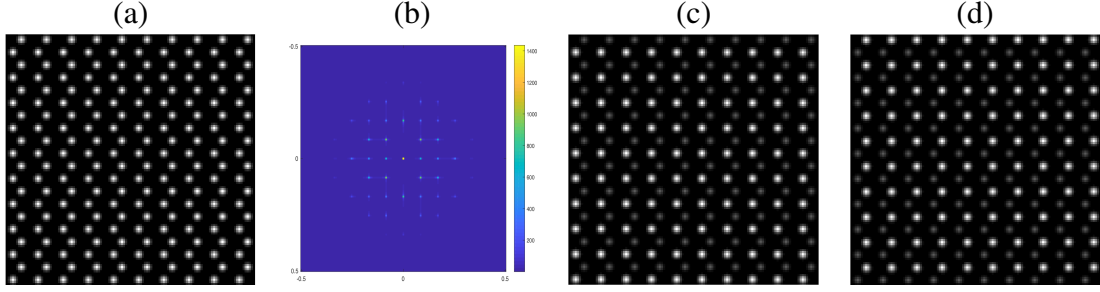


Figure 2.13: Effect of relative translation. (a) A superlattice composed of $\mathcal{T}_{4-3i}\Lambda\langle 12, i \rangle$ and $\mathcal{T}_{-4+3i}\Lambda\langle 12, i \rangle$. (b) The power spectrum of (a) where peaks are missing due to the relative translations. From this incomplete reciprocal lattice, LISA identifies lattice (c) and (d) each shown in white, superposed over (a) in gray.

with the larger fundamental volumes are decreased. This coincides with our experience that dense lattices are easier to recognize compared to the sparse ones. Consequently, LISA tends to find lattices with smaller particle sizes and smaller fundamental volumes first, and it identifies the sparser ones later.

Relative translations of the lattice layers have a more delicate influence on the power spectrum surface. The translation in the spatial domain results in a phase change in the frequency domain, and it does not affect the power spectrum if there is only one lattice. When multiple lattices are superposed, the frequencies along one direction will interact with each other. Suppose for some $1 < m \leq N$, $\Lambda_1^*(\xi) = \dots = \Lambda_m^*(\xi) = 1$ and $\Lambda_j^*(\xi) = 0$ for $j = m+1, \dots, N$, then $\hat{U}(\xi)$ is a sum of m complex numbers, whose magnitude varies based on the directions of μ_1, \dots, μ_m . An extreme case is that, if $\Lambda_1 = \Lambda_2$, $\mu_1 = -\mu_2 \neq 0$, and there exists an ξ such that $\Lambda_1^*(\xi) = 1$ and $\text{Re}(\xi\bar{\mu}_1) \neq 0$, then $|\hat{U}(\xi)| = 0$. Figure 2.13 shows an example. LISA detects potential lattices, even though the reciprocal lattices are incomplete and the reciprocal bases are not minimal. If any basis of the reciprocal lattice remains high response in the power spectrum, LISA will consider it as a candidate to be evaluated.

2.5.4 Robustness of LISA against Gaussian Perturbation

In practice, the atomic configuration in a crystal-melt interface [162] can be modeled using a lattice distorted by a Gaussian perturbation. We modify Equation 2.22 to consider such cases. To simplify notations, we assume that there is only one unshifted lattice. It is easy to extend to multiple lattices with arbitrary translations. Ignoring the remainder image in Equation 2.1, the image \tilde{U} of a lattice $\mathcal{T}_0\Lambda(b_1, b_2)$ with perturbed particles can be expressed as

$$\tilde{U}(x, y) = \sum_{k_1, k_2 \in \mathbb{Z}} G_\sigma * \delta(k_1 b_1 + k_2 b_2 + \Delta x_{k_1, k_2} + i \Delta y_{k_1, k_2} - x - iy) ,$$

with $(\Delta x_{k_1, k_2}, \Delta y_{k_1, k_2}) \in \mathbb{R}^2$, denoting the perturbation on the particle parameterized by $(k_1, k_2) \in \mathbb{Z}^2$ in the lattice. The Fourier transform of \tilde{U} is

$$\hat{G}_\sigma(\xi) \sum_{k_1, k_2 \in \mathbb{Z}} \exp(-2\pi i \phi_{k_1, k_2}(\xi)) ,$$

where $\phi_{k_1, k_2}(\xi) = (\Delta x_{k_1, k_2} + i \Delta y_{k_1, k_2} + k_1 b_1 + k_2 b_2) \cdot \xi$. We assume that the perturbations are independent and identically distributed Gaussian vectors with uncorrelated coordinates, that is $(\Delta x_{k_1, k_2}, \Delta y_{k_1, k_2}) \sim \mathcal{N}(0, \Sigma)$ where $\Sigma = \begin{bmatrix} s^2 & 0 \\ 0 & s^2 \end{bmatrix}$, $s > 0$ constant, for any $(k_1, k_2) \in \mathbb{Z}^2$. This implies that for any ξ in the frequency domain,

$$\phi_{k_1, k_2}(\xi) \sim \mathcal{N}((k_1 b_1 + k_2 b_2) \cdot \xi, s^2 |\xi|^2) .$$

Some observations are immediate. First, for a single lattice, perturbations only alter the phases. If there are multiple lattices, the magnitude of the power spectrum will be modified as discussed in subsection 2.5.3. Second, $\mathbb{E}[\phi_{k_1, k_2}(\xi)]$ depends on the angle between $k_1 b_1 + k_2 b_2$ and ξ . The perturbations have stronger effects on non-lattice points than lattice points. If ξ is reciprocal to the lattice point $k_1 b_1 + k_2 b_2$, then they are perpendicular; thus, the

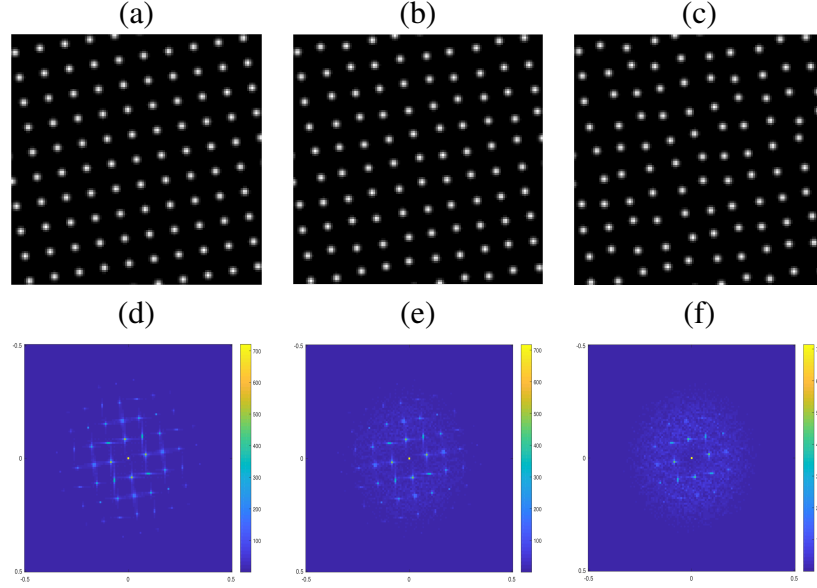


Figure 2.14: LISA's robustness against Gaussian perturbation. In the first column, a single lattice $\mathcal{T}_0\Lambda\langle 12, e^{i\pi/18} \rangle$ is shown in (a) with its power spectrum surface in (d). A centered Gaussian perturbation is applied with standard deviation (b) $s = 0.5$ and (c) $s = 1$, and their power spectra are displayed in (e) and (f), respectively. Notice that in the frequency domain, the reciprocal bases away from the origin are smeared by noises, but those near the origin remain high responses. The lattices identified by LISA in (b) and (c) are robust against the perturbation; their distances to (a) are 0.0046 and 0.0081, respectively.

average perturbation is 0. Finally, with a fixed s , the standard deviation of $\phi_{k_1, k_2}(\xi)$ only depends on $|\xi|$. When we are approximating relatively long periods, i.e., $|\xi|$ is small, the Fourier transform of the perturbed lattice is almost the same as that of the unperturbed one. In Figure 2.14, the lattice points are shifted by Gaussian perturbations with different standard deviations, and the low-frequency components maintain high responses. LISA is empirically robust against Gaussian perturbation with bounded standard deviation, and the detection of medium-sized lattices is effective. When the standard deviation is large, LISA identifies the correct lattices, yet the extraction procedure may be modified. For example, instead of direct subtraction, extract the point from the original image nearest to the candidate lattice.

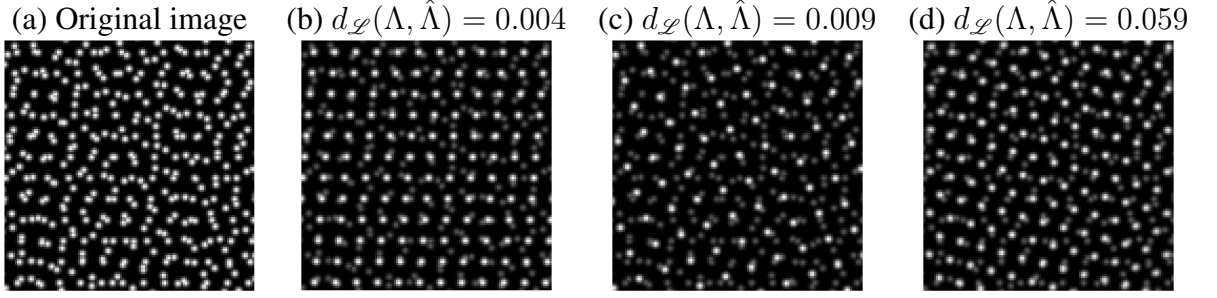


Figure 2.15: A typical example of LISA. (a) A superlattice of three lattices: $\mathcal{T}_{2-4i}\Lambda\langle -9.9927 + 0.0315i, 1.0014e^{i17\pi/36} \rangle$, $\mathcal{T}_{-7-4i}\Lambda\langle -4.4820 + 12.1815i, i \rangle$ and $\mathcal{T}_{1-5i}\Lambda\langle -4.9898 - 8.5389i, 1.0298e^{i7\pi/12} \rangle$. (b)–(d) display the lattices identified by LISA. Each metric value shows the distance between the true lattice and the identified one in \mathcal{L} .

2.5.5 Numerical Experiments with Various Superlattice Patterns

We present various numerical results in this section. The radius of each particle is set to be $2.5 \sim 3$ pixels, and we choose 2.7 for visualization. The performance of LISA is evaluated visually as well as numerically by computing the distances between the identified and the real patterns in the lattice space. The identified lattices are displayed in the same order as they are found during the iterations of LISA. For the choice of parameters, we fix $J = 6$ and $K = 10$.

Figure 2.15 shows a typical example of LISA. The given image is a superlattice composed of three distinguishable lattices, and LISA successfully extracts all the underlying lattices, one after another. For a better comparison, (b)–(d) display each identified lattice (white) overlaid on the original image (gray) (a). For each layer, the identified lattice and the true one show minor visual differences, which is also reflected in a small $d_{\mathcal{L}}$ value above the figure.

Figure 2.16 shows a more complex mixture where the given image (a) seems almost random. The more layers of lattices there are, the more complicated the superlattice becomes. Randomly clustered particles, curve-like segments, and highly inhomogeneous texton regions present visible challenges. As shown in (b)–(f), LISA identifies five different lattice patterns from (a) without any prior knowledge of the number of lattice layers, the lattice

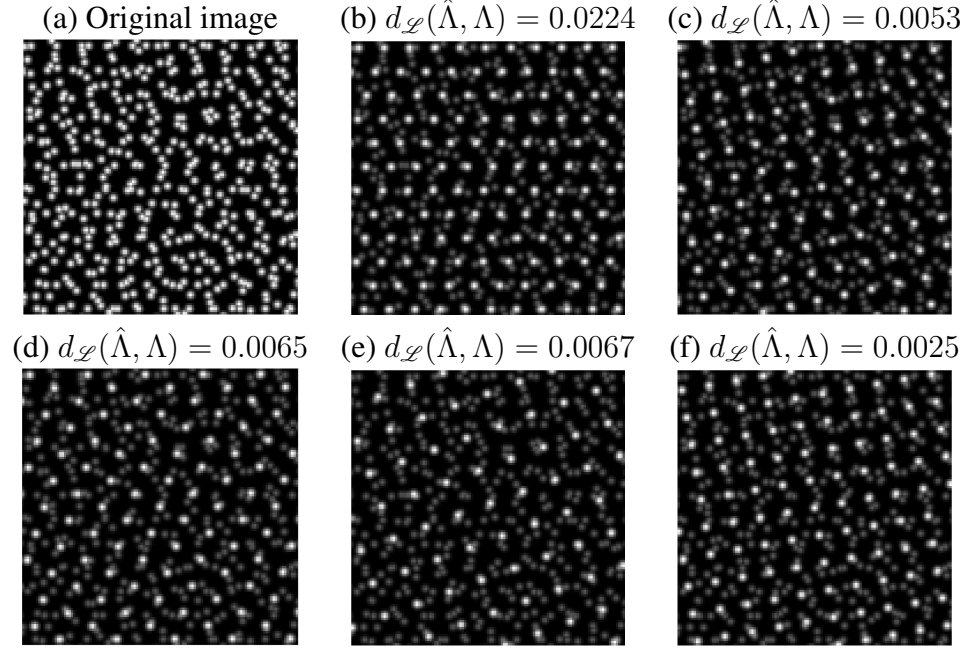


Figure 2.16: Superlattice with more layers. (a) A superlattice of 5 lattices: $\mathcal{T}_{2-5i}\Lambda\langle 11, e^{i7\pi/18} \rangle$, $\mathcal{T}_{3+4i}\Lambda\langle 11.7378 + 2.4949i, i \rangle$, $\mathcal{T}_0\Lambda\langle 3.7082 + 11.4127i, e^{4\pi/9} \rangle$, $\mathcal{T}_{1-2i}\Lambda\langle 14.0954 + 5.1303i, i \rangle$, and $\mathcal{T}_0\Lambda\langle 11.8177 + 2.0838i, i \rangle$. (b)–(f) show the extracted patterns using LISA. Notice that all the metric values $d_{\mathcal{S}}(\hat{\Lambda}, \Lambda)$ comparing the true lattices with the identified ones are very small.

translations, nor the lattice bases. The identified lattice patterns are of high precision. Their distances to the underlying true lattices are all less than 0.03. The identified lattice patterns (c) and (f) are very similar, and the distance between them in the lattice space is 0.0340. LISA can distinguish small differences since, in the power spectrum surface, the periodic structures are more easily identified as strong responses.

The new lattice representation and the metric are independent of the translation of a lattice pattern. Figure 2.17 presents the results of LISA concerning the translational lattices. There are four lattices mixed in the given image (a). They were generated with two different lattices, and each of them is translated differently to create an additional two distinct lattices. Using the cross-correlation function (in Step 2 of LISA), the underlying four lattices are extracted sequentially by LISA even if most of the particles are located close to each other. Figure 2.18 is a superlattice containing three lattices, generated with

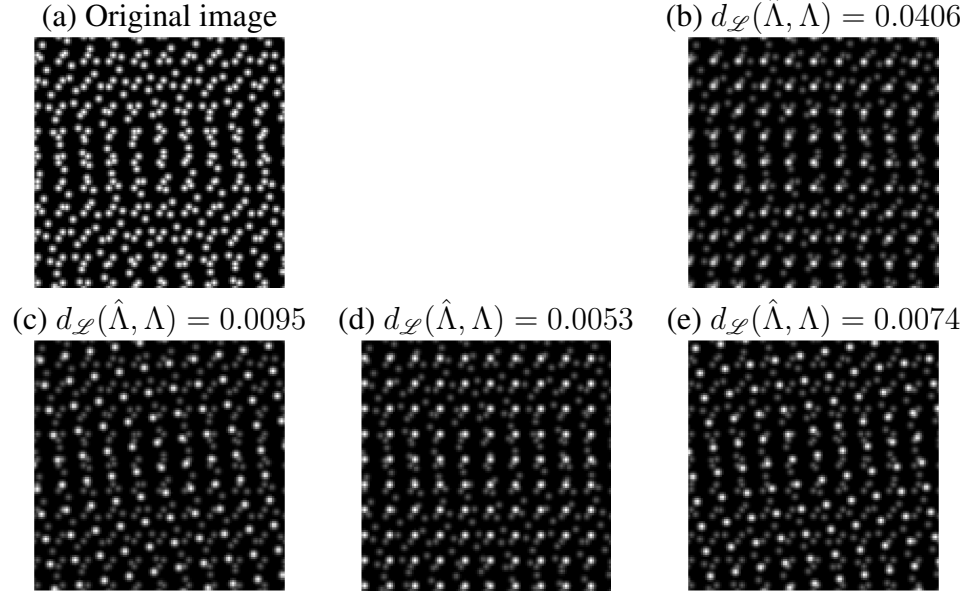


Figure 2.17: Mixture of translational lattices. (a) A superlattice of four lattices: $\mathcal{T}_0\Lambda\langle 12, i \rangle$, $\mathcal{T}_{1+i}\Lambda\langle 11.8177+2.0838i, i \rangle$, $\mathcal{T}_{2-3i}\Lambda\langle 12, i \rangle$, and $\mathcal{T}_{2-5i}\Lambda\langle 11.8177+2.0838i, i \rangle$. (b)–(e) show the identified patterns by LISA.

one lattice which is translated differently three times. Such a configuration results in many L-shapes [115] in the image. This local ambiguity presents no confusion for LISA since LISA observes the image globally in the frequency domain. The sensitivity of LISA to the distance between particles is affected by the particle size of the lattice candidates.

In practice, some images may contain partial lattice patterns. For example, Figure 2.19 (a) is a superlattice containing a complete lattice (b) and a partial lattice (c), where 50% of its particles are missing. The incompleteness modifies the power spectrum by convolving the reciprocal lattice of (c) with the Fourier transform of a lower triangular shape, resulting in weaker responses. LISA identifies the complete lattice (b) first, then reveals the incomplete lattice (c). Notice that LISA identifies the basis for the lattice pattern corresponding to (c), instead of the image of an incomplete lattice. This is shown in (e), where the identified pattern extends to the upper triangular region. (f) shows the intersection of the lattice identified in (e) (in white) with the original (a) (in gray). We also experiment in situations where 70% of the particles from one of the lattices are missing, and LISA recognizes the

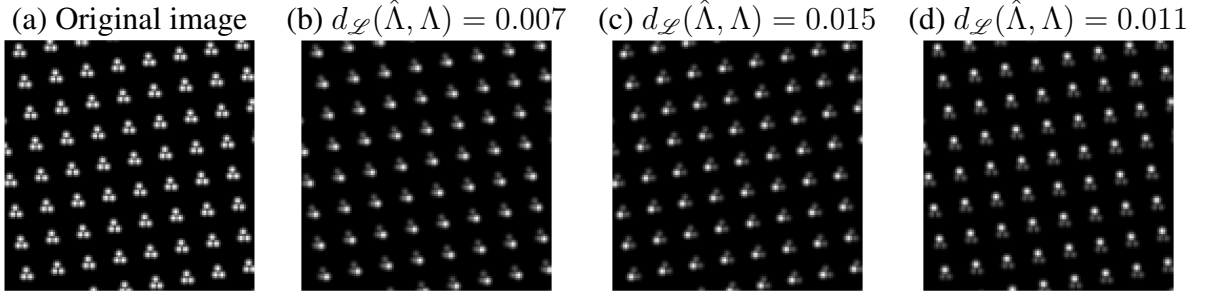


Figure 2.18: Close particles. (a) A superlattice of three lattices obtained by translating $\mathcal{T}_0\Lambda\langle 14.7721 + 2.6047i, i \rangle$ by $4 - 2i$, $1 - 2i$ and $2 - 5i$. These translations push particles close, and generate a pattern whose lattice points are composed of three dots. (b)-(d) show that LISA successfully distinguishes them with high precision as indicated by the values of $d_{\mathcal{L}}$.

incomplete lattices successfully. To be identified, the average intensity of the incomplete lattice must be at least 0.01, required by the terminating condition of LISA.

The evaluation energy (Equation 2.21) proposed in section 2.5 considers the density restriction, i.e., the overfitting term. Figure 2.20 illustrates its importance. Generated by two lattices, the superlattice in (a) presents a region of moiré pattern at the center. Without the density restriction in Equation 2.21, a dense lattice is identified as in (b); while with the density restriction, a different lattice is identified as in (c). The comparison between (b) and (c), as illustrated in (d), indicates that (c) is similar to a sub-lattice of (b). Although (b) has more points, many of them are not present in (a), which is demonstrated in (e). In this case, lattice (c) is visually indistinguishable from one of the underlying lattices. In the frequency domain, large-scale moiré patterns can produce strong responses on the power spectrum. Lattice candidates associated with these high responses are excessively dense, and they partially coincide with the moiré pattern in the given image, which causes instability for the subsequent identifications. The density restriction in Equation 2.21 makes LISA robust against possible moiré patterns.

Superposed lattices can present interesting patterns, and the formation may involve scaling and rotating. Hexagonal lattices, which share shape descriptors $\rho = \pm 1/2 + i\sqrt{3}/2$,

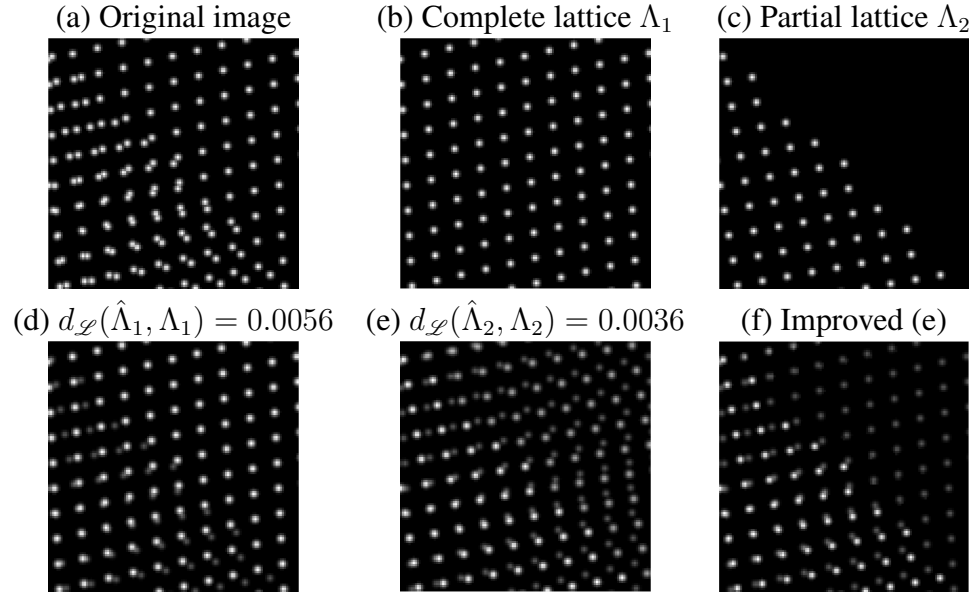


Figure 2.19: Incomplete lattice. (a) A superlattice composed of a complete lattice $\mathcal{T}_0\Lambda\langle 11.6924+2.6994i, e^{i4\pi/9}\rangle$, shown in (b), and a portion of $\mathcal{T}_{2-3i}\Lambda\langle 11.8177+2.0838i, i\rangle$, shown in (c). (d) and (e) are the identified patterns by LISA (in white) over the original (a) (in gray). (f) $\min(\mathcal{T}\hat{\Lambda}_2, I)$, where $\mathcal{T}\hat{\Lambda}_2$ is the identified lattice in (e) and I is the original image in (a). This shows the intersection of (a) and (e).

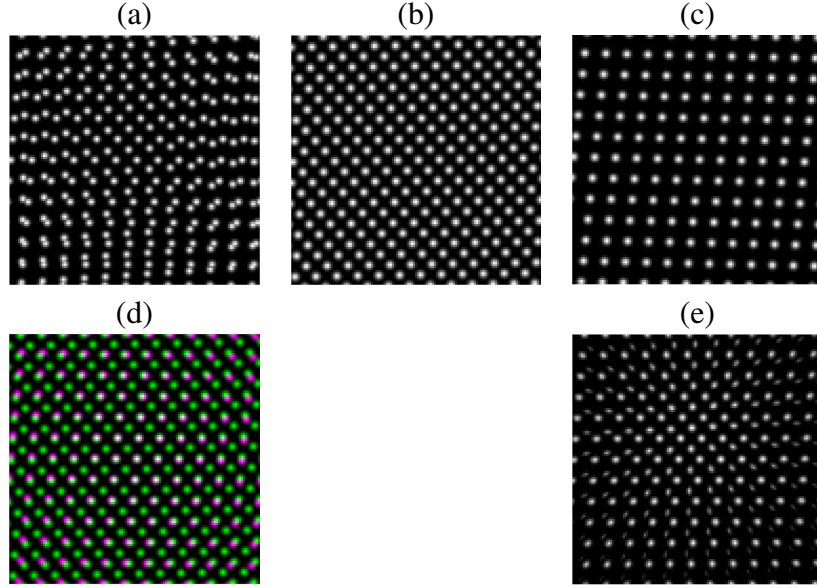


Figure 2.20: Importance of the density restriction. (a) A superlattice of $\mathcal{T}_{2-10i}\Lambda\langle 10, e^{i17\pi/36} \rangle$ and $\mathcal{T}_{-3+5i}\Lambda\langle 9.9756 + 0.6976i, e^{i17\pi/36} \rangle$. Without the second term in (Equation 2.21), we obtain a dense lattice $\mathcal{T}\tilde{\Lambda}$ in (b). With the density restriction, we get $\mathcal{T}\hat{\Lambda}$ in (c) which is the correct lattice pattern. (d) compares (b) and (c), where the white pixels are $\mathcal{T}\tilde{\Lambda} \cap \mathcal{T}\hat{\Lambda}$ (the particles commonly captured by (b) and (c)), the green are $\mathcal{T}\tilde{\Lambda} - \mathcal{T}\hat{\Lambda}$ (the extra points in (b) compared to (c)), and the red are $\mathcal{T}\hat{\Lambda} - \mathcal{T}\tilde{\Lambda}$ (particles in (c) not covered by (b)). It shows that (c) is almost a sub-lattice of (b). (e) $\min\{\mathcal{T}\tilde{\Lambda}, I\}$ showing the intersection of (a) and (b). This shows that the dense lattice (b) approximates the moiré pattern at the center of (a)

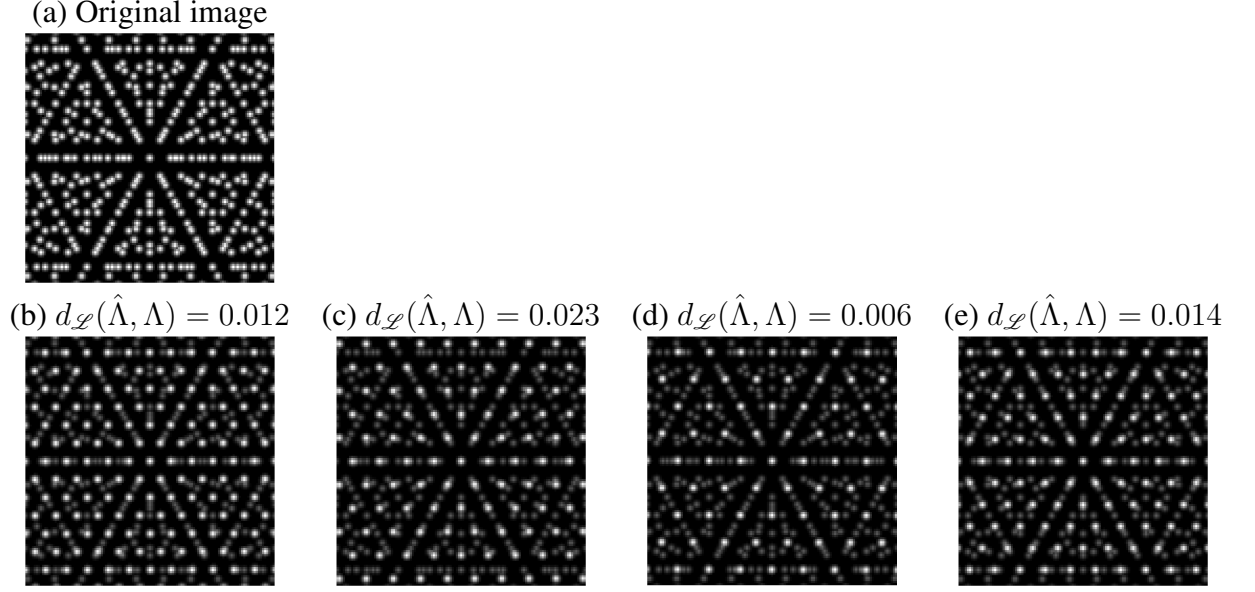


Figure 2.21: Flake-like pattern generated by lattices. (a) A flake-like superlattice of hexagonal lattices with β equal to 10, 13, 15 and 12. In the same order, (b)–(e) show LISA successfully identifies the underlying lattices.

are useful components; in the lattice space, their equivalent classes have the most elements, i.e., they are more symmetrical than the other lattices. In Figure 2.21, four hexagonal lattices with $\beta = 10, 12, 13$ and 15 are superposed, and the superlattice displays a flake-like pattern. In Figure 2.22, a flower-like pattern is formed by four hexagonal lattices with identical scale descriptor norm $|\beta| = 11$, and with different inclination angles: 53° , -53° , 143° and -143° . LISA successfully identifies each lattice pattern. We compare each identified lattice with the original one, and the small distance value $d_{\mathcal{L}}$ (above each figure) shows the effectiveness of LISA, even for complicated mixed patterns.

Figure 2.23 (a) displays a portion of an image from [163], which is acquired by performing SAED on a Na-exfoliated single-layer MoS_2 . A TMD monolayer contains three layers of lattices. In the top-view, S-atoms on the top overlap with those in the bottom. LISA successfully identifies the visible layers shown in (b). Figure 2.23 (c) shows a part of an HREM image of a MoSe_2 monolayer from [164]. In (d), underlying lattices with bright lattice particles are identified and separated by LISA, yet the dimmer lattice particles fail

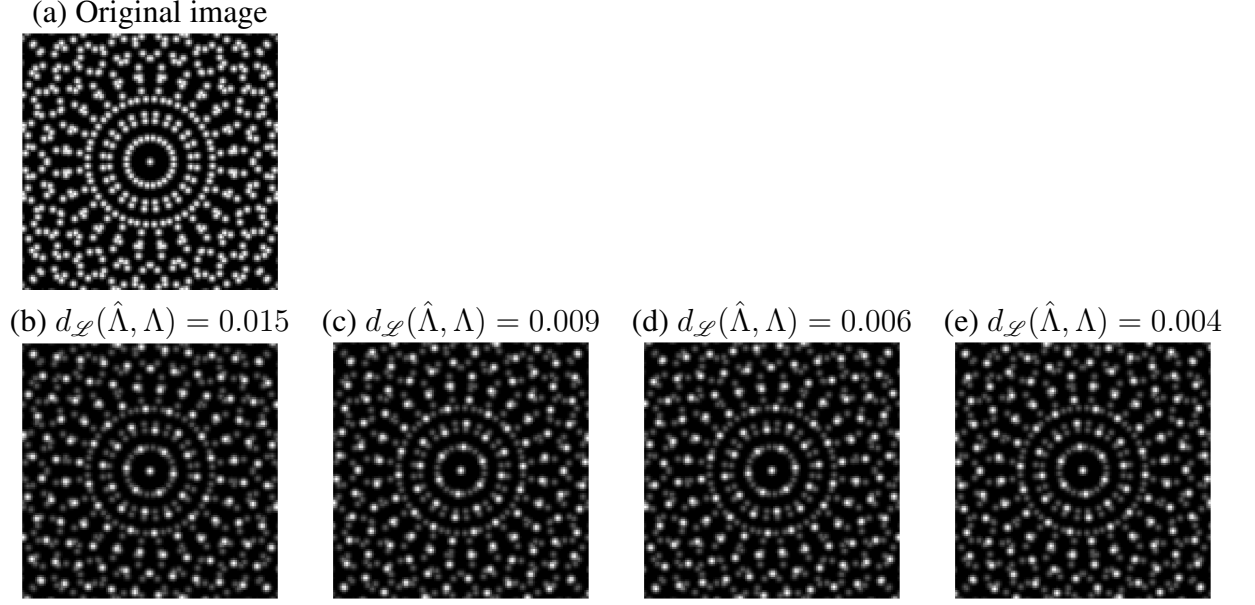


Figure 2.22: Flower pattern generated by lattices. (a) A flower superlattice of four lattices with scale descriptors having a common norm $|\beta| = 11$, and inclination angles equal to 53° , 143° , -53° and -143° . (b)–(e) show the lattices identified with high precision by LISA.

to be recognized. This can be addressed by lowering the threshold obtained by the Otsu’s method, image enhancing techniques, or sophisticated feature point detectors.

We test LISA on the grain segmentations from material science. LISA identifies one lattice pattern from each homogeneous region. By directly comparing these identified lattices with the preprocessed given image, we classify the grain regions. Different from the grain boundary detection, here we focus on the classification of the lattice patterns, which is similar to [165]. Figure 2.24 (a) shows a part of an image from [166], where a grain boundary is formed in the graphene grown by chemical vapor deposition (CVD). LISA detects two lattices as in (b) and (c). In (d), particles in the given image (a) shared with (b) are colored green, and those shared with (c) are colored red. The white particles indicate where (b) and (c) intersect. This example demonstrates the potential applications of LISA beyond superlattice separation.

Finally, we investigate the computational efficiency of LISA. We focus on three major

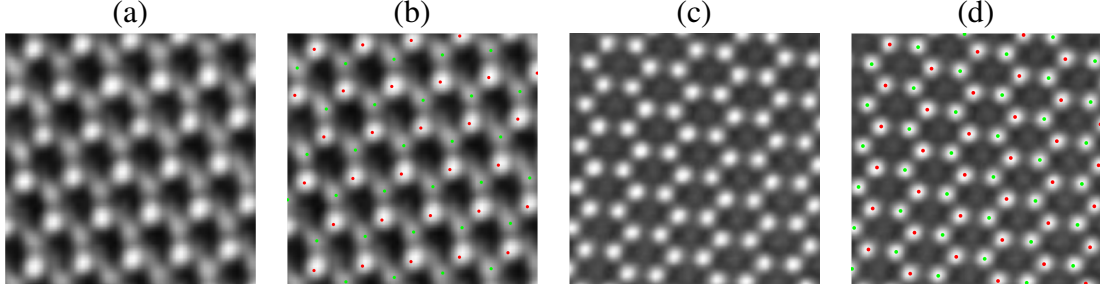


Figure 2.23: LISA on real images. (a) and (c) are the images of TMD monolayers adjusted from [163] 3 and [164] 1 (c), respectively. (b) and (d) show the identified lattice patterns, and lattice points from different layers are colored in red and green, respectively.

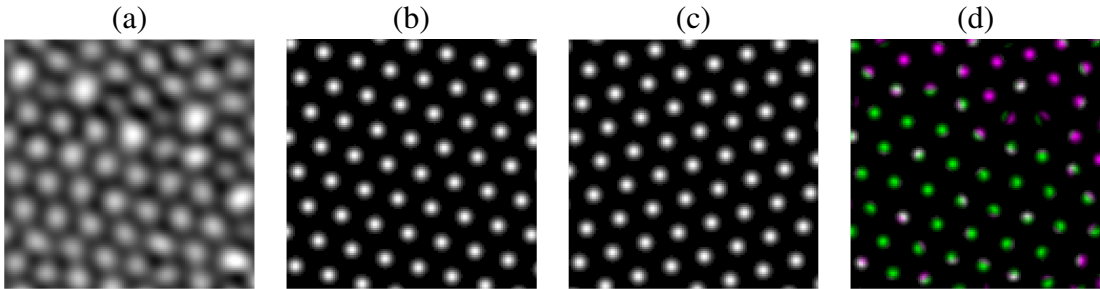


Figure 2.24: LISA on grain segmentation. (a) A grain image adjusted from [166] 15 (a). (b) $\mathcal{T}_{-1.3794+9.7510i}\Lambda\langle -10.9881 - 12.1163i, -0.4579 + 0.8950i \rangle$ and (c) $\mathcal{T}_{9.6287+9.5640i}\Lambda\langle -15.7326 - 4.7420i, 0.4813 + 0.8800i \rangle$ are the lattice patterns identified by LISA. (d) Particles shared in (a) and (b) are colored in green, and those shared with (c) in red. The white particles are shared by the lattices in (b) and (c).

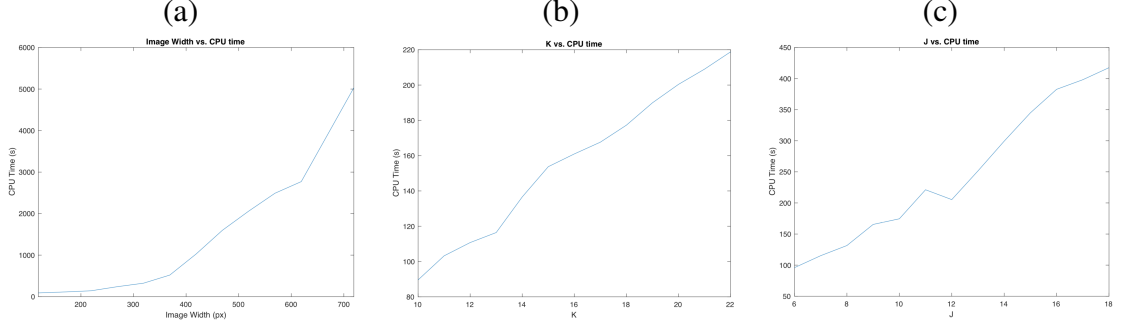


Figure 2.25: CPU time of LISA. Fixing two lattices, the base image width is $m = 119$, $K = 10$, and $J = 6$. (a) The image width m is increasing while K and J are fixed. (b) The number of iteration K is increasing with m and J fixed. (c) The number of connected components J is increasing while keeping m and K fixed. Roughly, LISA depends linearly on J and K respectively, and quadratically on m .

factors contributing to the run-time of LISA: the image size, the number of connected components on the spectrum surface (J in Algorithm algorithm 1), and the number of stabilizing iterations (K in Algorithm algorithm 1). Fixing a superlattice consisting of two lattices $\mathcal{T}_0\Lambda\langle 12, i \rangle$ and $\mathcal{T}_0\Lambda\langle 11.2763 + 4.1042i, e^{i4\pi/9} \rangle$, the CPU times (in seconds) of LISA are plotted against each one of these factors when the other two are fixed. The results roughly show that the complexity of LISA depends linearly on J and K , and quadratically on the image width. This is consistent with the analysis in [161], where the complexity of the B-spline convolution-based Radon transform is proportional to the image size, i.e., image width times image length.

2.6 Application to Grain Defect Detection

In crystalline materials, a grain is a homogeneous region that is composed of a single layer of crystal [167]. The presence of crystal defects such as particle dislocation, grain deformation, and grain boundary has unfavorable influences on macro-scale properties of the materials. To automatically detect these structures in atomic-scale 2D crystal images, many methods are developed in the literature. Variational model-based methods [102, 103, 104, 168, 105, 107] extract and classify the grains by minimizing certain functional en-

ergy; wavelet type methods [169, 170, 171, 106] measure the local properties of wave-like components to reveal crystal defects; and Voronoi type methods [172, 173, 174] detect the problematic particles by comparing the Voronoi cells with hexagonal polygons. Different from these methods, we approach the grain defect detection problem by clustering particles in an abstract metric space directly, where lattice representation is unique, and comparison between lattice patterns is systematic.

In this section, we apply the lattice metric space $(\mathcal{L}, d_{\mathcal{L}})$ developed in [118] to grain defect detection problem. The lattice metric space $(\mathcal{L}, d_{\mathcal{L}})$ can detect lattice inconsistencies such as grain boundaries in non-hexagonal crystalline materials without any particular modification. Voronoi type methods are limited to hexagonal lattices, and other techniques are involved to analyze non-hexagonal ones [174]. In 2D, the other types of lattices are oblique (e.g., orthoclase), square (e.g., halite), primitive rectangular (e.g., epsomite) and centered rectangular (e.g., hemimorphite). These classes are encoded in (β, ρ) and further refined by considering the metric structure on \mathcal{L} . Moreover, the metric $d_{\mathcal{L}}$ provides a single-valued yet comprehensive measurement of the dissimilarities between any grains. It is robust and sensitive enough to expose the grain boundaries and to reveal particle dislocations as well as continuous deformations.

Specifically, we describe an efficient algorithm to extract grains and detect grain defects. The main idea of this method is to classify particles into visually distinct grains by checking their vicinities. For each particle, by comparing a 9-point stencil with nearby points, a lattice is extracted and mapped to \mathcal{L} . In \mathcal{L} , Riemannian center of mass is non-unique, which poses an obstacle for application of clustering methods requiring the notion of centers, such as k-means [175]. To tackle this challenge, an over-segmentation using regularized k-means [176] is employed, and the clusters are merged \mathcal{L} with an agglomerative hierarchical clustering method [175]. In addition, we introduce a function considering the sizes of clusters to assist the cut-off selection for the merging.

Table 2.1: Lattice Clustering Algorithm

Inputs:

1. U : given gray-scale image;
2. λ : the parameter in regularized k-means;
3. T : threshold for merging.

Step 1. Particle and local lattice identification. Each local maxima is refined by fitting a narrow Gaussian to find each particle p_j , $j = 1, 2, \dots, N$. Among the k -nearest ($k = 5$) neighbors of each particle p_j , two vectors which gives the best match of a 9-point stencil is picked for p_j . Let \mathcal{A} be the collection of such vectors in \mathbb{R}^4 .

Step 2. Apply the regularized k-means to \mathcal{A} , and obtain K clusters $\{\mathcal{C}_l\}_{l=1}^K$ with Euclidean centers $\{(\bar{\beta}_l, \bar{\rho}_l)\}_{l=1}^K$.

Step 3. Considering the equivalence relations among descriptors, merge $\{\mathcal{C}_l\}_{l=1}^K$ by clustering the lattices $\{\Lambda\langle\bar{\beta}_l, \bar{\rho}_l\rangle\}_{l=1}^K$ in \mathcal{L} with hierarchical clustering using the threshold T .

2.6.1 Lattice Clustering Algorithm based on Lattice Metric Space

We introduce an efficient algorithm to capture structural defects in multigrain from a gray-scale image. Our approach is based on clustering, and particles with visually similar vicinity are grouped. The algorithm is summarized in Table 2.1.

In Step 1, each particle is located by identifying local maxima refined by fitting a narrow Gaussian. For each particle p_j , k -nearest ($k = 5$ in our method) particles are found, denoted as q_1, q_2, \dots, q_k . For each pair of (q_s, q_t) , $s \neq t$, we construct a 9-point stencil with Gaussian weight located at $p_j + a(q_s - p_j) + b(q_t - p_j)$, $a, b = 0, \pm 1$. We choose the pair with the highest response and compute its corresponding descriptors (β_j, ρ_j) . Then p_j is assigned with the lattice $\Lambda\langle\beta_j, \rho_j\rangle$.

In Step 2, we first cluster the lattices $\mathcal{A} = \{\Lambda\langle\beta_j, \rho_j\rangle\}_{j=1}^N$ by clustering the descriptors $\{(\beta_j, \rho_j)\}_{j=1}^N$ in \mathbb{R}^4 equipped with the Euclidean norm. As discussed in section 2.3, the quotient geometry of \mathcal{L} is non-trivial, which complicates directly clustering the descriptors in \mathcal{L} . To overcome this difficulty, we over-segment the lattices using the regularized k-means [176] in \mathbb{R}^4 first. The reason is that if (β_j, ρ_j) and (β_k, ρ_k) are close in \mathbb{R}^4 , then

$\Lambda\langle\beta_j, \rho_j\rangle$ and $\Lambda\langle\beta_k, \rho_k\rangle$ are close in \mathcal{L} , but the inverse is not true. Consequently, there might be two clusters whose members are similar lattices. We denote the resulting clusters by $\{\mathcal{C}_l\}_{l=1}^K$ with centers $\{\Lambda\langle\bar{\beta}_l, \bar{\rho}_l\rangle\}_{l=1}^K$, where $(\bar{\beta}_l, \bar{\rho}_l)$ is the Euclidean center of pairs of descriptors in \mathcal{C}_l .

In Step 3, we merge the clusters $\{\mathcal{C}_l\}_{l=1}^K$ considering the equivalence relations among lattices. We compute the pair-wise lattice distances $d_{\mathcal{L}}$ among $\{\Lambda\langle\bar{\beta}_l, \bar{\rho}_l\rangle\}_{l=1}^K$ and employ the standard agglomerative hierarchical clustering method [177]. This method requires a cut-off $t > 0$, and cluster centers closer than t will be merged. To assist choosing the optimal threshold, we consider the following function:

$$g(t) = \max_{m=1,2,\dots,s} \left\{ \frac{\sum_{j=J_{m-1}+1}^{J_m} |\mathcal{C}_j|}{\max_{j=J_{m-1}+1,\dots,J_m} \{|\mathcal{C}_j|\}} \right\}, \quad (2.23)$$

which measures the energy of the current stage of clustering using cut-off t ; here, without loss of generality, $J_0 = 0$, $\{\mathcal{C}_j\}_{j=1}^{J_1}$, $\{\mathcal{C}_j\}_{j=J_1+1}^{J_2}$, \dots , and $\{\mathcal{C}_j\}_{j=J_{s-1}+1}^{J_s}$, represent the clusters of particles to be merged respectively when the cut-off is t ; and $|\mathcal{C}_l|$ denotes the counting measure $l = 1, 2, \dots, K$. In general, $g(t)$ is a staircase function of $t > 0$. In order for the merging to be *stable* and *substantial*, we pick optimal thresholds T in the intervals upon which the graph of g is flat and the previous discontinuity has high jump.

2.6.2 Numerical Experiments on Grain Defect Detection

We apply LCA to images from the literature [102, 104, 178, 179, 180]. Throughout this paper, we fix the weighting parameter $w = 0.05$, and the suitable range for λ in regularized k-means is found to be $0.2 \sim 0.5$. The particles belonging to a common grain region are colored with identical color.

Lattice representation: (b_1, b_2) versus (β, ρ)

Compared to basis representation (b_1, b_2) , descriptor representation (β, ρ) is stable. $\text{Arg } b_2$ ranges from 0 to 2π , yet $\text{Arg } \rho$ takes value in $[\pi/3, 2\pi/3]$, which is more restrictive. In practice, grains in polycrystalline materials generally have similar lattice patterns, thus the shape descriptors ρ in the data are concentrated. The benefit of \mathcal{L} is exemplified in Figure 2.26. Fixing a particle, colored red, as the reference, we compare the lattice representations (b_1, b_2) and (β, ρ) to those at the reference. Darker color indicates closer in the chosen distance. If we compare (b_1, b_2) in Euclidean metric for \mathbb{R}^4 , then (d) and (f) display the results for two distinct reference points which are nearly random. Using the same reference points, if we apply the lattice metric $d_{\mathcal{L}}$ on (β, ρ) , (e) and (g) show the advantage of using descriptor representation compared to (d) and (f) respectively. The color distributions in (e) and (g) comply with our perception of homogeneity. Figure 2.26 (e) and (g) also reveal grain defects and continuous deformation within homogeneous regions.

We note this experiment is done by comparing every particle to the reference (red point) without using LCA. Every lattice label is compared to that of the reference particle directly in \mathcal{L} , and the distance is represented by the color. This approach shows excellent results, yet the computation is very slow. In the following, we apply LCA, which is more computationally efficient.

General example of LCA

There are two parameters in LCA: λ and T . λ is a penalty parameter in the regularized k-means, which implicitly controls the number of clusters K . In general, choosing λ is straightforward [176], and the optimal ones are found within wide intervals. Large λ penalizes clusters with small elements, hence fewer clusters with even sizes are obtained; small λ produces more clusters with even sizes. Applying LCA to the image in Figure 2.26 (a), we plot the function $g(t)$ in Figure 2.27 (a) when t ranges from 0.1 to 1. Notice that $g(t)$ is a staircase function of t , and the figure shows clear plateau within intervals, e.g.

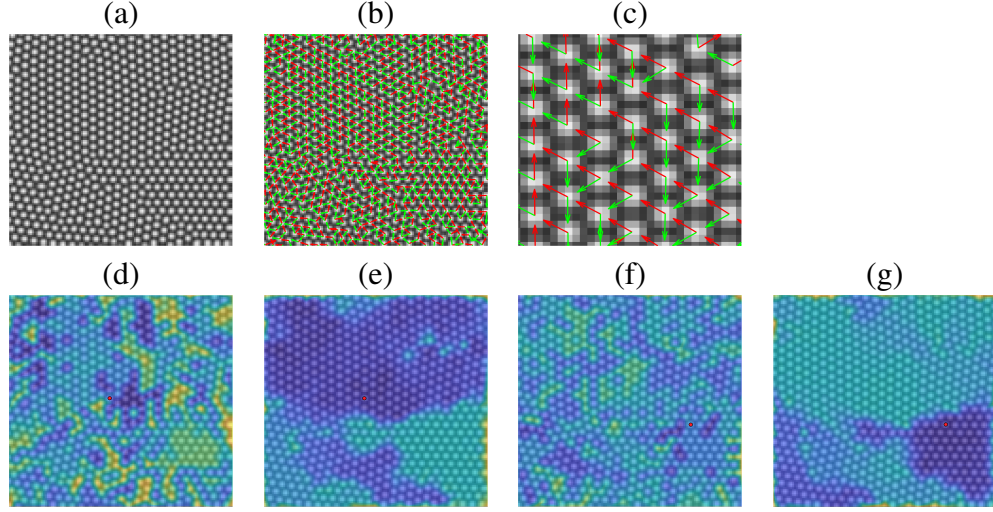


Figure 2.26: Direct classification using $d_{\mathcal{L}}$. (a) PFC image from [104] Fig. 4. (b) Lattice labels obtained in Step 1 of LCA. (c) zoomed-in partial region from (b). (d) and (f) show Euclidean distance function of (b_1, b_2) representation with respect to that of the red points. (e) and (g) show the lattice distance function $d_{\mathcal{L}}$ of (β, ρ) representations with respect to that of the red points, which are the same as those in (d) and (f) respectively. Linear interpolation is applied to fill the color in (d)–(g).

$[0.25, 0.35]$, $[0.35, 0.45]$, $[0.47, 0.7]$. One should avoid choosing the threshold T near the jump-discontinuities. Any perturbation near these points will produce substantially different results. The results with T from the flat regions $T = 0.4$, $T = 0.5$ and $T = 0.8$ present different levels of details, as shown in (b)–(d) in order. Compared to (c), (b) distinguishes minor disorientation such as those between red and green regions. While large scale boundaries remain noticeable in (c), only particles inconsistent with ambient patterns are accentuated in (d). Typical particle defects are not identified in region based methods, e.g., compare Figure 2.26 with 5 of [104]. Otherwise, both methods agree on large scale grain boundaries. We also note that regions labeled with the same color have similar lattice pattern, and they can be separated in different locations, for example, the green regions in (b).

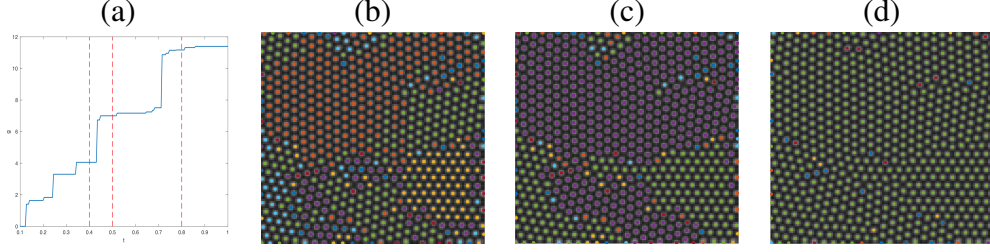


Figure 2.27: Apply LCA to the image Figure 2.26(a). Here (a) shows the curve $g(t)$. Results when (b) $T = 0.4$, (c) $T = 0.5$ and (d) $T = 0.8$ show the effect of T .

Over-segmentation using k-means versus Regularized k-means

In LCA, we choose the regularized k-means to over-segment. Using regularized k-means, the number of clusters K is implicitly controlled by the parameter λ , and K dynamically evolves depending on the data. This is desirable, especially when we do not know a suitable K a priori. Moreover, regularized k-means is stable. Every step during the clustering is deterministic, hence the results from regularized k-means are reproducible. To justify our choice, we experiment and compare the regularized k-means with the most common clustering method, k-means [175]. First, K has a direct impact on the results, yet it is not straightforward to choose. If we apply the k-means instead of the regularized k-means in LCA Step 1, fix $T = 0.5$, and rerun the algorithm for 100 times for different choices of the number of clusters K , then Figure 2.28(a) shows that the number of identified grains clearly depends on K . Second, the random initialization in k-means causes difficulties in finding a stable threshold T . Even with the same parameters, e.g., $K = 30, T = 0.5$, it exhibits different results each time we run the algorithm, as shown in (b) and (c). Notice that the blue region in (b) is connected, yet the corresponding purple region in (c) is not. Also, the point defects in the upper right within the grains have different features in these results. In (d), using $K = 50$ and the same threshold $T = 0.5$ produces a result similar to Figure 2.28 (c), yet this is not reproducible. However, all the results in Figure 2.27 are deterministic in the sense that, for any given image, the same combination of λ and T yields

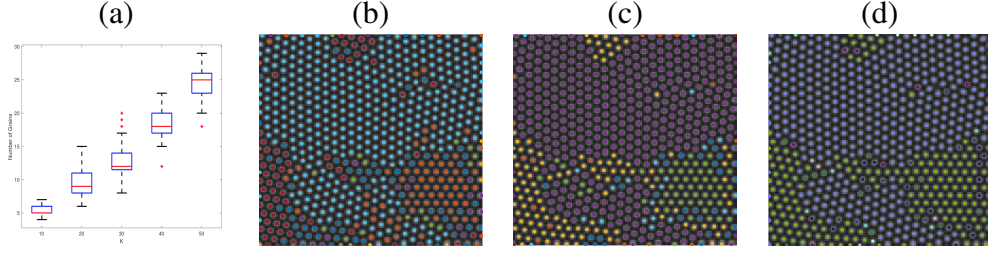


Figure 2.28: Instability of k-means. (a) Box-plot of the number of grains against parameter K in k-means. (b)–(d) use k-means with different initializations. (b) and (c) set $K = 30$ and $T = 0.5$ and (d) uses $K = 50$ and $T = 0.5$.

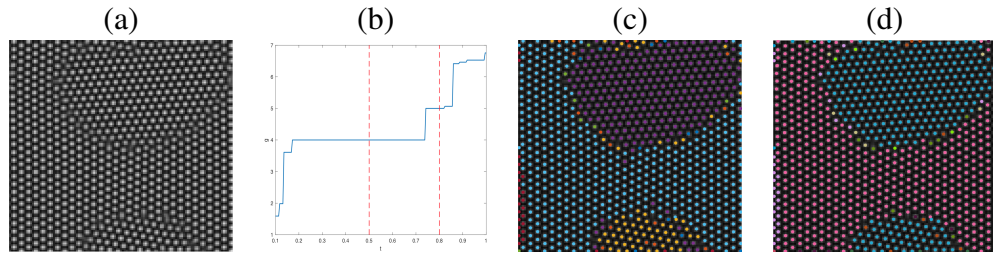


Figure 2.29: (a) There are 3 grains: one on the top, one in the middle, and one in the bottom. Image from [102] Fig. 1. (b) The curve $g(t)$ with 3 major jump-discontinuities. (c) $T = 0.5$. (d) $T = 0.8$.

an identical result.

Graph of g and grain features.

The domain for the function g defined in Equation 2.23 is closely related to the lattice metric $d_{\mathcal{L}}$, and the graph of g reveals geometrical features about the multigrain. In Figure 2.29 (a), we apply LCA, and g is plotted in (b). Observe that there are 3 major jumps in the graph of g , which exactly correspond to the grain regions in (a). The grain in the bottom of (c) is merged with the top one when $T = 0.5$ in the plateau after the first jump is changed to $T = 0.8$ after the second jump, as shown in (d). It is also clear that the height of each jump is proportionally related to the grain area. See Figure 2.30 and Figure 2.31 for images with two grain regions. Notice that the threshold T are chosen on the plateau of the function g respectively.

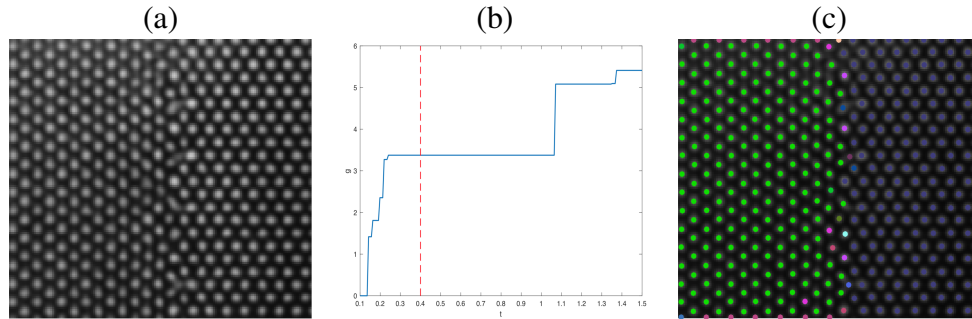


Figure 2.30: (a) There are 2 grains with a regular boundary. Image adapted from [180] Fig. 1(a). (b) The curve $g(t)$ with 2 major jump-discontinuities. (c) Result with $T = 0.4$.

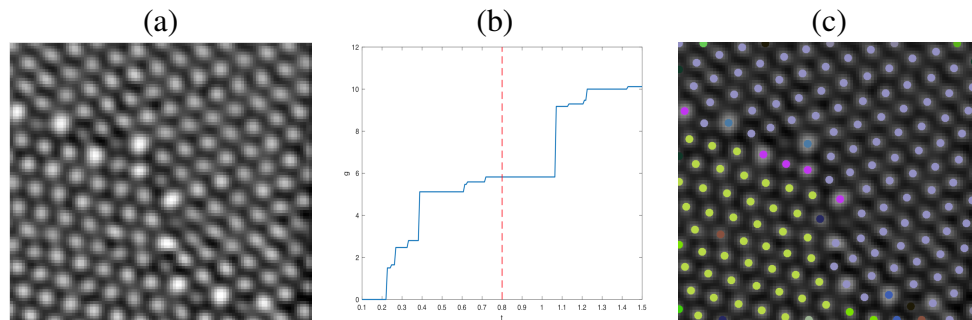


Figure 2.31: (a) There are 2 grains presented and the grain boundary is irregular. Image adapted from [178] Fig. 1. (b) The curve $g(t)$ with 2 major jump-discontinuities and the jump is rough. (c) Result with $T = 0.8$.

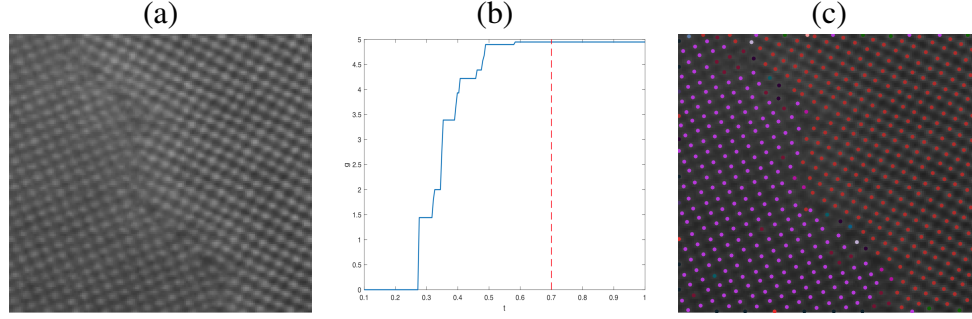


Figure 2.32: (a) Grain boundary between non-hexagonal grains. Image adapted from [179] Fig. 3. (b) The curve $g(t)$. (c) Result with $T = 0.7$.

Behavior of LCA near grain boundaries.

In Figure 2.29(c), Figure 2.30 (c) and Figure 2.31 (c), LCA assigns the boundary particles to either of the neighboring lattice patterns; in other cases, LCA creates new classes. This differs from grain boundary identification using variational approaches, where spatial constraints such as length minimization are added. In the framework of lattice metric space, each particle is classified only based on k -nearest points, thus it is free to create new clusters.

Grain boundary detection in non-hexagonal crystalline materials.

As we discussed in Section 2, the generality of the lattice metric space theory allows analysis applicable to any types of Bravais lattices. Since LCA is established on this framework, we can directly employ LCA to images composed by arbitrary types of grains. Figure 2.32 (a) is a HAADF-STEM image presenting a grain boundary in well-annealed, body-centered cubic (BCC) Fe. The graph of g in (b) implies choices for T approximately greater than 0.6, and the result with $T = 0.7$ is shown in (c). Due to fewer symmetries than the hexagonal lattices, the boundary between two cubic lattices has a relatively larger gap; hence it is challenging for methods that depend on hexagonal cells. LCA, with the flexibility supported by the lattice metric space, classifies the particles on the edges of the gap correctly.

2.7 Summary

This chapter addresses two questions about lattice identification and separation in superlattices. The first one is: *What is a proper space where we can compare any two lattices quantitatively?* Starting from the positive minimal bases, we exploit the modular group theory and the Poincaré metric to define a lattice space \mathcal{L} with a natural metric structure. This new definition provides rich geometrical intuition for the collection of lattices. The computation of the metric $d_{\mathcal{L}}$ yields compact and visually consistent measure, well-representing differences between lattice patterns. While it is compatible with the wallpaper group theory, \mathcal{L} provides finer classification.

The second question is: *How to practically identify and separate lattices from a superlattice?* We introduce the algorithm LISA. Without prior knowledge of the number of lattice layers in the superposed lattices, LISA sequentially identifies and extracts individual lattice patterns until the remainder image has insufficient intensity. We show the importance of density restriction when evaluating the lattice candidates indicated by pairs of high responses on the power spectrum surface. This evaluation method renders LISA's robustness against moiré patterns. An analytical framework is presented to explore the effects of relative translations and Gaussian perturbations. The metric space $(\mathcal{L}, d_{\mathcal{L}})$ allows more discussion about special families of lattices, and its geometrical properties are interesting to explore. LISA detects regular lattice patterns as well as near-regular lattices, and it can be extended to the identification of grain defects.

We also demonstrated an application of the lattice metric space theory to grain defect detection problems. Most methods such as [104, 168] are free from particle position estimation, since they focus on homogeneous regions. We detect inconsistencies in the local patterns based on positions of k-nearest neighbors of each particle. More sophisticated techniques can be applied during this preprocessing stage. Theoretically, this approach is easily linked with the lattice metric space theory, which provides a uniform framework to

classify particles in materials of any type of 2D Bravais lattice. In the described lattice clustering algorithm, LCA, different from variational methods, we emphasize the neighboring similarity of each particle rather than large scale homogeneity; thus LCA is superior at identifying various grain defects including grain boundaries. Since length minimization is not applicable, isolated particle deviating from the lattice point of the neighboring pattern will create a new cluster.

CHAPTER 3

PDE-BASED SHAPE REPRESENTATION AND VECTORIZATION

Shape representation plays an important role in feature analysis [181], object recognition [182] as well as classification [183]. A 2D shape is represented either by contour or region, and both can be further classified as structural or global [47]. For example, the chain code [48] is a contour-based structural method; the Fourier descriptor [49] is a contour-based global method; the convex hull is a region-based structure method; and the Zernike moment [50] is a region-based global method. These representations provide compact information applicable for different purposes. An attractive approach is to encode the shape using simple geometries such as points, curves, or polygons. It allows user-friendly shape manipulation and scalable image rendition [184]. In this chapter, we focus on two types of representations: shape skeleton and silhouette vectorization.

The skeleton of a 2D shape [185, 186, 187, 188, 189, 190] is a region-based structural representation method. We discuss the flux-ordered thinning algorithm proposed in [191] and refer to it as the *Hamilton-Jacobi Skeleton* (HJS). The first part of the algorithm is to compute the distance function from the boundary of the shape using a Hamiltonian formalism of the Eikonal equation. Then the flux is defined at each point using the gradient vector field of the distance function. In the second part, each boundary point with high flux is examined and removed if the homotopy of the shape remains unchanged. This homotopy preserving thinning procedure continues until no point is removable anymore, then these points constitute the skeleton of the shape.

Silhouette vectorization, on the other hand, is a contour-based structural approach. A silhouette is a subset of the plane traditionally obtained by copying on paper the shadow projected on a wall by a person placed in front of a point light source¹. In digital images,

¹<https://en.wikipedia.org/wiki/Silhouette>

an object can be delineated by a mere luminance threshold (e.g., Otsu’s algorithm [192]) as long as it is darker or brighter than its surrounding. After a mere color quantization, many graphics software² reduces the image to a piecewise constant image, i.e., a union of disjoint 2D shapes. The boundaries of these shapes are encoded in the Scalable Vector Graphics (SVG) format³ where each involved primitive element is specified by a small number of 2D vectors, called *control points*, and the encoded shape can be scaled independently from the resolution. Such conversion from pixel image to SVG is called *vectorization*.

3.1 Region-based Representation – Shape Skeleton

Let $I : \Omega \rightarrow \{0, 1\}$ be a binary image, where $\Omega = [0, M] \times [0, N]$ denotes the continuous image domain, and M, N are positive integers. A 2D shape, A , is a subset of Ω with piecewise analytical boundary. Its *skeleton*, S , is a set of points which has equal shortest distance from two or more boundary points of A [185]. More precisely, the skeleton of a 2D shape is a finite union of C^2 curves that are either closed or ending at a finite set of junctions or endpoints [193]. One of the many equivalent definitions of skeleton [186, 187, 188, 189, 190] is based on the distance transform: the skeleton S of a shape A consists of the singularities of the distance transform restricted to the inside of A [191]. For example, the skeleton of a disk is its center point, and the skeleton of a square is given by its diagonals. Figure 3.1 shows the skeletons of some simple shapes. A skeleton can be used to reconstruct a shape via the medial axis transform [186]. It defines a coordinate system placed along the skeleton that encodes the distance from a skeleton point to the boundary of the shape. Hence, the contour can be recovered as the envelope of a series of circles centered at the skeleton with radii specified by the distance transform.

In the literature, different methods have been devised to skeletonize 2D shapes from various perspectives. For example, Voronoi diagram based methods [194, 195] focus on characterizing the symmetry axis; continuous transformation based approaches [196, 191,

²See (e.g.) https://en.wikipedia.org/wiki/Adobe_Illustrator or Vector Magic

³https://fr.wikipedia.org/wiki/Scalable_Vector_Graphics

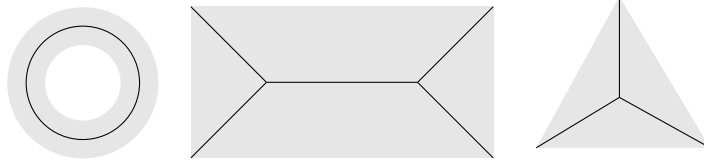


Figure 3.1: Skeletons (black curves) of some elementary shapes.

197, 186, 188, 185, 189, 190] aim at extracting the singularity set of certain evolution of the boundary curve; and most morphology-based techniques [198, 199] look for maximally inscribed balls whose centers compose the skeleton. We refer the readers to [200] for a detailed review. A modified U-net was developed to directly map 2D shapes to their respective skeletons [201].

3.2 Hamilton-Jacobi Skeleton Algorithm

As summarized in [191], HJS has two main parts: Part I, distance function and flux computation, and Part II, homotopy preserving algorithm. We present the details in four subsections. For Part I, the distance function is computed in subsection 3.2.1. We propose to use the fast sweeping algorithm [202, 203] for the distance computation. In subsection 3.2.2, the flux is defined for each point using the gradient field of the distance function. For Part II, in subsection 3.2.3, the homotopy preserving point classification is explained, and in subsection 3.2.4, the flux-ordered thinning algorithm is presented.

3.2.1 Distance Transform using the Fast Sweeping Algorithm

Let $I : \Omega \cap \mathbb{N}^2 \rightarrow \{0, 1\}$ be a discretized binary image, where $\Omega \cap \mathbb{N}^2$ represents a union of square pixels on a grid. We denote by $\text{Int}(\Omega) \cap \mathbb{N}^2 = \{1, 2, \dots, M-1\} \times \{1, 2, \dots, N-1\}$ the set of pixels in the interior of the image domain. On the continuous domain, the distance map $D : \Omega \rightarrow \mathbb{R}$ to the contour ∂A of the shape A is defined by

$$D(x, y) = \min_{(x', y') \in \partial A} \sqrt{(x - x')^2 + (y - y')^2}. \quad (3.1)$$

The distance D to the contour ∂A is the *unique* viscosity solution [204, 205] of the Eikonal equation

$$\begin{cases} |\nabla D(x, y)| = 1 , \\ D(x, y) = 0 , (x, y) \in \partial A . \end{cases} \quad (3.2)$$

This is a first-order nonlinear PDE of Hamilton-Jacobi type which does not have classical solutions. To solve D from Equation 3.2, we employ the *fast sweeping algorithm* [203, 202]. The fast sweeping algorithm is an iterative method that uses an upwind scheme for discretization and Gauss-Seidel iterations with alternating sweeping orders for solving the Eikonal equations [202].

In the discrete setting, a pixel $(i, j) \in \Omega \cap \mathbb{N}^2$ is contained in the given shape A if $I[i, j] = 0$, and outside of A if $I[i, j] = 1$. To trace the contour of A on the discrete domain $\Omega \cap \mathbb{N}^2$, a pixel $(i, j) \in \text{Int}(\Omega) \cap \mathbb{N}^2$ is considered a *boundary point* if it satisfies:

1. $I[i, j] = 0$, and
2. at least one of its 8-neighbors is outside the shape, i.e., $I[i \pm 1, j \pm 1] = 1$.

We denote by \mathcal{B} the set of discrete boundary points.

On a rectangular grid, we discretize Equation 3.2 using the Godunov upwind difference scheme for the interior points $(i, j) \in \text{Int}(\Omega) \cap \mathbb{N}^2$:

$$[(D[i, j] - D_{x, \min})^+]^2 + [(D[i, j] - D_{y, \min})^+]^2 = 1 , \quad (3.3)$$

where $D_{x, \min} = \min\{D[i + 1, j], D[i - 1, j]\}$, $D_{y, \min} = \min\{D[i, j + 1], D[i, j - 1]\}$, and $(z)^+ = z$ if $z > 0$ and $(z)^+ = 0$ otherwise. For pixels on the edge of the image, an one-sided difference scheme is applied. We initialize $D[i, j] = 0$ for every $(i, j) \in \mathcal{B}$ and assign large positive values for the others. Next, we sweep the whole computational

domain with four alternating orderings:

- 1) $i = 0, 1, \dots, M, j = 0, 1, \dots, N.$
- 2) $i = M, M - 1, \dots, 0, j = 0, 1, \dots, N.$
- 3) $i = M, M - 1, \dots, 0, j = N, N - 1, \dots, 0.$
- 4) $i = 0, 1, \dots, M, j = N, N - 1, \dots, 0.$

As we are at pixel $(i, j) \in \Omega \cap \mathbb{N}^2$ during the sweeping, we compute the solution $\overline{D}[i, j]$ of Equation 3.3 by

$$\overline{D}[i, j] = \begin{cases} \min\{D_{x,\min}, D_{y,\min}\} + 1, & |D_{x,\min} - D_{y,\min}| \geq 1, \\ \frac{1}{2}(D_{x,\min} + D_{y,\min} + \sqrt{2 - (D_{x,\min} - D_{y,\min})^2}), & |D_{x,\min} - D_{y,\min}| < 1. \end{cases} \quad (3.4)$$

Then we update $D[i, j]$ with $\min\{\overline{D}[i, j], D[i, j]\}$. Upon completing the four specified sweepings, the fast sweeping algorithm terminates; hence, the total computational cost is $O(MN)$. In practice, we only need to update the values of the pixels in the interior of the shape.

In [202], Zhao proved that the iterative solution by the fast sweeping algorithm converges monotonically to the solution of the discretized system (Equation 3.3). After four iterations, the iterative solution at every pixel is bounded from above by its true distance. Since the numerical Hamiltonian (Equation 3.3) is monotone and first-order consistent, combining with the fact that the numerical solution from a monotone and consistent scheme converges to the viscosity solution [205], Zhao concluded that the solution from the fast sweeping algorithm converges to the distance map. Moreover, by providing a pointwise error estimation, Zhao proved that this solution is optimal in the sense that any other method solving Equation 3.3 has the same accuracy if not worse. These properties hold in n -dimensional Euclidean space ($n \geq 1$) as well, where the fast sweeping algorithm takes 2^n iterations.

Remark Starting from the continuous PDE setting, specifically, the Eikonal equa-

tion (Equation 3.2), we focus on the fast-sweeping algorithm for its efficiency and simplicity as a PDE numerical scheme. Thanks to one of the reviewers of this paper, we would like to acknowledge the advances in Euclidean Distance Transform (EDT) in discrete graph settings. In [206], the authors gave a survey comparing six algorithms proposed between 1994 and 2003. In 2012, Felzenszwalb and Huttenlocher [207] proposed a simple method focusing on the cost function defined on a grid. They formulated the problem as the minimum convolution of two functions and proposed a simple and fast algorithm. We compare the performance in subsection 3.3.4.

Input: I a binary image where a pixel is 0 if it is inside the shape and 1 otherwise.

Compute the set of boundary points \mathcal{B} .

Assign $D[i', j'] = 0$ for all $(i', j') \in \mathcal{B}$, and $D[i', j'] = M^2 + N^2$ for all $(i', j') \in (\Omega \cap \mathbb{N}^2) \setminus \mathcal{B}$.

for $i=0, 1, \dots, M-1, j=0, 1, \dots, N-1$ **do**

if $i = M - 1$ **then**

 Define $D_{x,\min} = D[i - 1, j]$.

end

else if $i = 0$ **then**

 Define $D_{x,\min} = D[i + 1, j]$.

end

else

 Define $D_{x,\min} = \min\{D[i - 1, j], D[i + 1, j]\}$.

end

 Define $D_{y,\min}$ along the y -direction similarly.

 Compute $\bar{D}[i, j]$ according to Equation 3.4.

 Update $D[i, j] = \min\{\bar{D}[i, j], D[i, j]\}$.

end

Repeat the above procedure for the other 3 sweeping directions, i.e.,

$i = M, M - 1, \dots, 0, j = 0, 1, \dots, N$.

$i = M, M - 1, \dots, 0, j = N, N - 1, \dots, 0$.

$i = 0, 1, \dots, M, j = N, N - 1, \dots, 0$.

Output: Distance map D for the interior points of the shape in I .

Algorithm 2: Distance Computation: The Fast Sweeping [202]

3.2.2 Computation of Average Outward Flux

The skeleton points can be distinguished from the others by comparing their average outward fluxes derived from the gradient of the distance transform. On the continuous domain Ω , the average outward flux F of ∇D is defined by the outward flux through the boundary of its neighboring region R , normalized by the Hausdorff measure of ∂R :

$$F(x, y) = \frac{\int_{\partial R} \langle \nabla D, \mathcal{N} \rangle ds}{|\partial R|}, \quad (x, y) \in \Omega \quad (3.5)$$

where \mathcal{N} is the outward normal along ∂R , and ds is the length element. By the assumption that the boundary of a 2D shape is piecewise analytical (in fact, being smooth suffices), the divergence of ∇D can be regarded as a measure supported by the skeleton of the shape. By the divergence theorem, when (x, y) is not a skeleton point, and R is sufficiently small, $F(x, y) = 0$. At any skeleton point which is not a crossing, up to a translation and a rotation, the distance function D admits a local expansion

$$D(x, y) \approx D_0 - x \text{ for } x \geq 0; \quad D(x, y) \approx D_0 + x \text{ for } x < 0$$

where the skeleton point is placed at $(0, 0)$, and D_0 denotes the distance from $(0, 0)$ to the shape's boundary.

If we consider the disk $R_r(s, 0) \equiv \{(x, y) : (x - s)^2 + y^2 \leq r^2\}$, with $r > 0$ then the average outward flux of ∇D is given by

$$\frac{\int_{\partial R_r(s, 0)} \langle \nabla D, N \rangle ds}{|\partial R_r(s, 0)|} = \begin{cases} \frac{-\int_{-\arccos(s/r)}^{\arccos(s/r)} \cos(t) r dt + \int_{\arccos(s/r)}^{2\pi - \arccos(s/r)} \cos(t) r dt}{2\pi r} = -\frac{2}{\pi} \sqrt{1 - \left(\frac{s}{r}\right)^2} & \text{if } |s| < r \\ 0 & \text{if } |s| \geq r \end{cases}$$

therefore the average outward flux of ∇D has a minimum at the skeleton point when we move in the direction orthogonal to the skeleton curve and the function $s \rightarrow -\frac{2}{\pi} \sqrt{1 - \left(\frac{s}{r}\right)^2}$

gives us an idea about how the average outward flux varies when approaching the skeleton.

More generally, notice that for any nonnegative test function $\varphi \in C_c^\infty$ with a sufficiently small support R such that $(0, 0) \in R$, we have

$$\begin{aligned} \int_{\partial R} \langle \nabla D, \nabla \varphi \rangle ds &= \iint_{\mathbb{R}^2} \operatorname{div}(\nabla D) \varphi dx dy \\ &\approx - \iint_{x \leq 0} \varphi_x dx dy - \iint_{x \geq 0} -\varphi_x dx dy \\ &= -2 \int_{\mathbb{R}} \varphi(0, y) dy . \end{aligned}$$

The above calculation explains why, numerically, skeleton points are expected to have a clearly negative outward flux.

We compute the gradient vector field $\nabla D = (\partial_x D, \partial_y D)$ using a finite difference scheme that can be derived from optimization. We approximate $D(x, y)$ using a linear function:

$$\widehat{D}(x, y) = D[i, j] + a(x - i) + b(y - j) .$$

Comparing to the Taylor expansion, we have $\partial_x D[i, j] \approx a$ and $\partial_y D[i, j] \approx b$. To determine the coefficients a and b explicitly, we fit \widehat{D} to the 3×3 stencil centered at $P = (i, j) \in \operatorname{Int}(\Omega) \cap \mathbb{N}^2$ by minimizing the following weighted squared-error:

$$\begin{aligned} \mathcal{E}(a, b) &= (D[i+1, j] - \widehat{D}(i+1, j))^2 + (D[i-1, j] - \widehat{D}(i-1, j))^2 + \\ &\quad (D[i, j+1] - \widehat{D}(i, j+1))^2 + (D[i, j-1] - \widehat{D}(i, j-1))^2 + \\ &\quad \frac{1}{\sqrt{2}}(D[i+1, j+1] - \widehat{D}(i+1, j+1))^2 + \frac{1}{\sqrt{2}}(D[i-1, j-1] - \widehat{D}(i-1, j-1))^2 + \\ &\quad \frac{1}{\sqrt{2}}(D[i+1, j-1] - \widehat{D}(i+1, j-1))^2 + \frac{1}{\sqrt{2}}(D[i-1, j+1] - \widehat{D}(i-1, j+1))^2 . \end{aligned}$$

This provides approximations for partial derivatives at interior points:

$$\partial_x D[i, j] = (1 - \alpha) \frac{D[i+1, j] - D[i-1, j]}{2} + \alpha \frac{D[i+1, j+1] - D[i-1, j+1] + D[i+1, j-1] - D[i-1, j-1]}{4} \quad (3.6)$$

$$\partial_y D[i, j] = (1 - \alpha) \frac{D[i, j+1] - D[i, j-1]}{2} + \alpha \frac{D[i+1, j+1] - D[i+1, j-1] + D[i-1, j+1] - D[i-1, j-1]}{4} \quad (3.7)$$

where $\alpha = 2 - \sqrt{2}$. Note that the scalar α is derived from requiring the estimated gradient to have a norm invariant under rotations of 45° .

In the discrete domain, to compute $F[i, j]$ as defined in Equation 3.5 for the pixel $(i, j) \in \text{Int}(\Omega) \cap \mathbb{N}^2$, we replace R with a square containing the 8-neighboring points of (i, j) . We pre-compute the outward normals at the neighboring points, $\mathcal{N}_n = (\mathcal{N}_n^x, \mathcal{N}_n^y)$, $n = 0, 1, \dots, 7$, as illustrated in Figure 3.2 (a), and we approximate the integral by a Riemann sum. The formula for computing the average outward flux at (i, j) is:

$$F[i, j] = \frac{1}{8} \sum_{n=0}^7 (\partial_x D[i, j] \times \mathcal{N}_n^x + \partial_y D[i, j] \times \mathcal{N}_n^y) . \quad (3.8)$$

3.2.3 Point Classification based on Local Topology

As a thinning algorithm, HJS finds the skeleton of a shape by consecutively removing non-skeleton points. For an appropriate shrinkage, each pixel needs to be examined carefully based on the local topology, i.e., the intensity distribution of its 8-neighboring pixels. In particular, two types of points are critical in the success of HJS.

A point is *simple* if its removal does not affect the topology of the object [191]. This means that removing a simple point does not create a new connected component nor a hole in the original shape. We construct a graph G for every pixel $P \in \text{Int}(\Omega) \cap \mathbb{N}^2$ based on

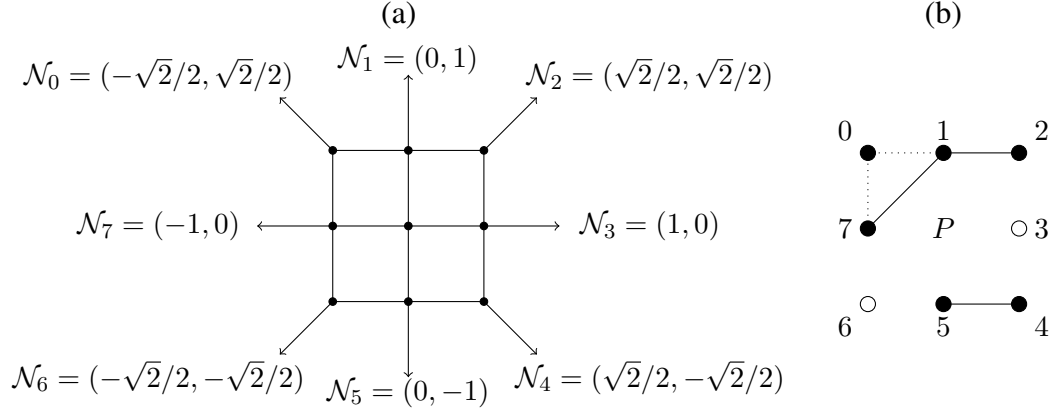


Figure 3.2: (a) The normal vectors at the neighboring points used for approximating the flux Equation 3.8 at the central pixel. (b) An example graph G constructed for the pixel P . For any arbitrary pixel, its 8 neighborhoods are indexed as shown here. Neighboring pixels inside the shape (black circles) are the vertices of G , and two vertices are connected if they are 8-neighborhood to each other. We avoid the 3-loops at the corners, e.g., $0 - 1 - 7$, by directly connecting the furthest two among them.

its 8-neighbors and use the Euler's characteristic of a graph to decide if P is simple. See Figure 3.2 (b). In G , each vertex corresponds to a neighboring pixel that is inside the shape A , and there is an edge if the associated two pixels are neighbors to each other. Because P is simple if and only if G is a tree [191], it suffices to check if the number of vertices minus the number of edges of G , i.e., the Euler characteristic of graph, is exactly 1. For convenience, we label the 8-neighboring pixels by integer indices from 0 to 7 in a clockwise orientation starting at the top-left one (see Figure 3.2 (b)).

The procedure for checking if a point is simple goes as follows. At each point, construct a set V collecting the indices of neighbors that are inside A . Initialize $v = 0$ and $e = 0$ for recording the number of vertices and edges of G , respectively. For $k = 0, 1, \dots, 7$, if both neighbor k and neighbor $\text{mod}(k + 1, 8)$ are found in V , we increase both v and e by 1; otherwise, we increase only v by 1. The graph may contain unnecessary loops of length 3 when all the three vertices on the corners are in V . For example, the neighbor 0, 1, and 7 in Figure 3.2 (b). To simplify the graph, we delete the shortest two edges in such a loop. In particular, if the neighbor k ($k = 0, 2, 4, 6$) is in V , and if both neighbor $\text{mod}(k - 1, 8)$ and

neighbor $\text{mod}(k + 1, 8)$ are in V as well, we reduce both v and e by 1. Finally, we compute $v - e$; if it is 1, then we mark P as a simple point, otherwise, P is not a simple point.

In addition, endpoints need to be tracked. An *endpoint* corresponds to the end of a 4-connected or 8-connected digital curve [191]. Identifying these points helps to produce robust skeletons and avoid interior points in the final result. A pixel P is an endpoint if the associated vertex set V only has one element, or if V only has two elements whose indices differ by 1 or 7, i.e., they are 4-neighborhood to each other. Otherwise, P is not an endpoint.

3.2.4 Homotopy Preserving Thinning

The second part of HJS, called the homotopy preserving thinning, sifts through the pixels inside the shape such that the remaining pixels are the skeleton points. The flux computed in subsection 3.2.2 as well as the point type discussed in subsection 3.2.3 are the keys. To avoid early removals of the pixels that are likely to be skeleton points, the average outward flux ranks the pixels inside the shape from high to low, which are then examined in order. To preserve the homotopy of the shape, only simple points and endpoints with high fluxes are considered removable.

An important part of the implementation is the heap data structure, which allows operations such as `insert`, `top`, and `pop`. Elements stored in a heap are associated with sorting keys. A heap will automatically sort the inserted data so that the first element, retrieved by `top`, always has the maximal key value; and `pop` automatically deletes the top element. It is noted that a heap only allows access to the top element, thus retrieving the element with the maximal key is very efficient. In C++, one may define a heap via a `priority_queue`; and in Python, one can resort to `heapq`.

In our case, we construct a heap of pixels with their average outward fluxes as the sorting keys. To save the storage, we directly update the pixels of the given binary image I using three types of labels: points that are currently in the heap (label 2), removed points

(label 1), and candidate skeleton points (label 0).

First, we insert all the boundary points that are simple into a heap H sorted by their fluxes and label them by 2. We sift through the points via iterations. During the updates, some points that are not simple may become simple later, or vice versa; thus it is necessary to test simple points at each iteration. In each iteration, we extract the top element in the heap by applying $H.\text{top}()$ followed by $H.\text{pop}()$. If the top element is an endpoint and its average flux is below a threshold, τ , indicating that it has more potential to be a skeleton point (See subsection 3.2.2), we remove it from H and update its label by 0. Otherwise, we change its label to 1, then we consider its 8-neighbors. If any of them is labeled 0, we insert it in the heap together with its sorting key and label it by 2. Here we propose a practical formula for the threshold:

$$\tau = \min_{(i,j) \in A \cap \mathbb{N}^2} F[i, j] / \gamma . \quad (3.9)$$

This $\gamma > 0$ is a parameter whose default value is $\gamma = 2.5$ in our experiments. The iteration terminates when there are no more points to be removed, i.e., H is empty. The pixels labeled by 0 constitute the skeleton of the shape A .

We display a complete description of HJS in the form of pseudo-code in algorithm 3. The full algorithm divides into two parts. Part I computes the distance function and the average outward flux of the gradient of the distance transform, and Part II describes the homotopy preserving thinning.

3.3 Numerical Experiments on Shape Skeletons

We present numerical results to illustrate various interesting aspects of HJS. Throughout the experiments, our default choice was $\gamma = 2.5$. If the image is grayscale with dynamic range $[0, 255]$, we transform it into binary by setting the pixels to be 1 if the intensity > 125 , and 0 if the intensity ≤ 125 . For an RGB image, we use the same binarization on

Input: I a binary image which takes value 0 for points inside the shape, and 1 for outside; τ a threshold parameter for flux value.

Part I: Distance Function and Average Outward Flux

Compute the distance map D Equation 3.11 of the set of boundary points.

Compute the gradient vector field ∇D according to Equation 3.6 and Equation 3.7.

Compute the average outward flux of the gradient, F , using Equation 3.8.

Part II: Homotopy Preserving Flux-ordered Thinning

Initialize an empty heap H . For each point P on the boundary of the shape :

if P is simple **then**

 insert $(P, F(P))$ into a heap H with the flux $F(P)$ as the sorting key;

 update $I(P) = 2$

end

while H is not empty **do**

 Let $(P, F(P)) \leftarrow H.\text{top}()$. Delete P from H via $H.\text{pop}()$

if P is simple **then**

if P is not an endpoint **OR** $F(P) > \tau$ **then**

 update $I(P) = 1$

for neighboring point Q of P **do**

if $I(Q) = 0$ **then**

if Q is simple **then**

 Insert $(Q, F(Q))$ into H

 update $I(Q) = 2$.

end

end

end

end

else

 update $I(P) = 0$

end

end

end

Output: A binary image I where pixels labeled by 0 represent skeleton points.

Algorithm 3: Hamilton-Jacobi Skeleton (HJS) [191]

its lightness obtained via $(R + G + B)/3$.

3.3.1 Skeletonization of 2D Shapes

In Figure 3.3, we show the skeletons computed using HJS for various types of shapes. In the first row, variations from a disk shape are shown. Compared to a disk whose skeleton is a single point, these shapes have more complicated skeletons due to continuous modifications on their boundaries. Some features of the shape can be understood from the graph topology of the skeleton. For example, each convex corner of the shape creates a branch; the last shape in the first row of Figure 3.3 has 16 convex corners, and its skeleton is a tree with 16 branches. In the second row, we applied HJS to shapes of some common objects: a *cup*, an *apple*, a *vase*, and a *hammer*. Both the *cup* and the *vase* which are topologically equivalent to an annulus, provide examples for the fact that a skeleton may contain loops, the number of which is equal to the genus of the shape. We also observe the correspondence between the convex corners of the shape and the branches of the skeleton as mentioned above. This exact property explains instability in computing the skeleton: any perturbation on the boundary of the shape may create a prominent change in the skeleton. Applying HJS to varying shapes of the lizard in the third row, we observe some extraneous branches on the skeleton due to the non-smooth boundaries. For this issue, many techniques, such as skeleton evolution [208] and pruning [209], have been proposed to automatically delete the irrelevant branches.

3.3.2 Effects of the Parameter γ

In HJS, an endpoint is kept if its average flux is greater than a threshold τ . As a consequence of the homotopy preserving, the whole branch attached to that endpoint remains as a part of the skeleton. Therefore, when we increase γ in Equation 3.9, we expect to see more branches.

In Figure 3.4, we demonstrate the effects of γ by applying HJS to a shape modified

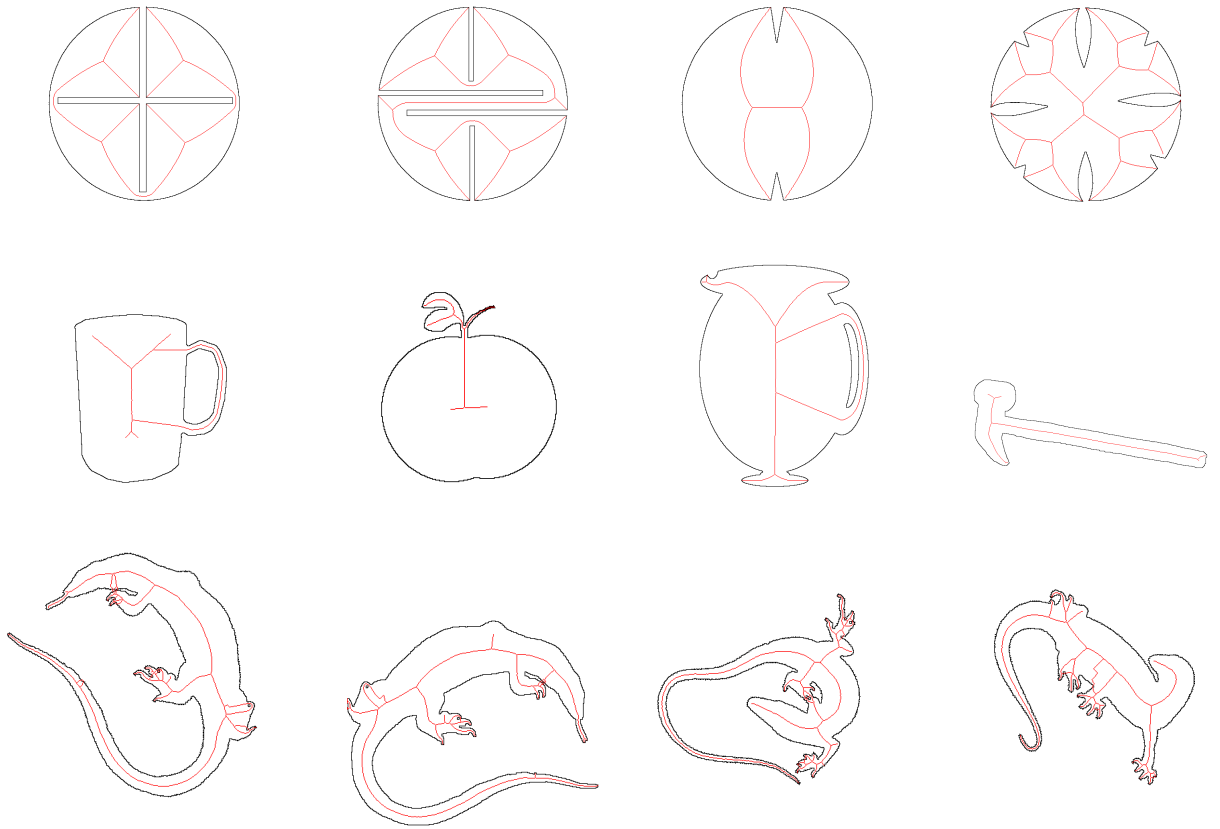


Figure 3.3: Skeletons (red curves) for various shapes computed by HJS. In all examples above, we used the default parameter $\gamma = 2.5$.

from a pentagon whose boundary is highly perturbed. Figure 3.4 (a) is produced when $\gamma = 2.5$ (the default choice in this paper); due to the irregularities on the boundary, many branches are created. Observe the length of the skeleton branches in contrast to the size of the perturbations on the boundary: they are not proportional. The local distribution of skeleton branches, rather than individual ones, may infer the smoothness of the boundary. For instance in (a), along a segment of the skeleton near the top-left boundary of the pentagon, there are more branches on the lower side compared to the upper side; within that range, the upper-side boundary is smoother than the lower-side boundary.

In (b), with $\gamma = 1.5$, many of the branches in (a) disappear, leaving only those associated with sharp corners. The order of cancellation is determined by the flux. During the homotopy preserving thinning, the threshold τ only acts on endpoints. Consequently, as long as the average flux at the tip of skeleton is comparatively low, the whole branch is kept as a part of the identified skeleton.

We further reduced γ to 1.2 in (c) and observe that only two major components remain: the S-shape and a single branch caused by the sharp corner at the bottom-left of the pentagon. By considering the medial axis to reconstruct the shape from the skeleton, with a small γ , one can obtain a compact representation of a shape with regularized contour. The evolution from (a) to (b), then to (c) shows that the set of skeletons extracted from different γ 's provides a multi-scale representation of the given shape. It is analogous to applying bandpass filters to a signal to separate the low-frequency components of the original sequence from the high-frequency components such as noise. As we gradually decrease γ , we omit the small-scale variations along the boundary, and focus more on capturing the principal shape. A similar approach to shape analysis can be found in [49, 210].

For a smaller value such as $0 < \gamma < 1$, HJS shows an interesting property. *The shape is simply-connected if and only if the skeleton identified using $\gamma < 1$ is a single point.* This is based on the fact that HJS preserves the homotopy. In Figure 3.5, we applied HJS with $\gamma = 0.9$ to three different shapes and used this property to check the simple-connectedness.

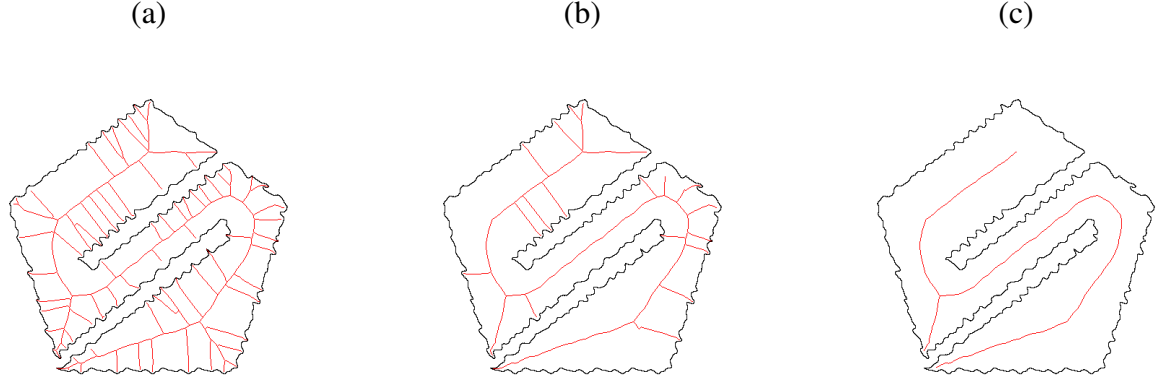


Figure 3.4: Multi-scale representation of the shape using skeletons computed by different γ . (a) $\gamma = 2.5$. (b) $\gamma = 1.5$. (c) $\gamma = 1.2$. By choosing a smaller γ , the identified skeleton becomes more robust against boundary perturbation and captures the large-scale shape features.

In (a), since the skeleton is a single point (shown at the corner of the right-bottom branch), the shape is simply-connected. The converged skeleton point appears at the minimum of the average fluxes of the entire image, and depending on the shape, it is not necessarily at the center of mass of the shape. In (b), the skeleton is homeomorphic to a circle, formed by the line segments (the *mug* body) and an arc (the handle). Since genus is a homeomorphic invariant, we infer that this *mug* shape has genus 1. It is worth noting that the number of connected components of the skeleton graph is the same as that of the shape. In (c), the skeleton consists of 10 points, hence the shape has 10 simply-connected components. With the remark that the skeletons identified using any $\gamma \in (0, 1)$ are identical, this experiment shows that HJS with $0 < \gamma < 1$ can be applied as an effective homotopy type detector.

Since HJS with $0 < \gamma < 1$ effectively distinguishes simply-connected shapes from the others, we apply it to detect small holes inside the shape which are not immediately visible to humans. Figure 3.6 (a) shows the non-trivial skeleton identified using $\gamma = 0.9$, which indicates that the perturbed boundary contains loops. When we zoom in the top-left (b) and bottom-right (c) of the original shape, the homotopy type of the shape is indeed modified by the perturbed pixels.

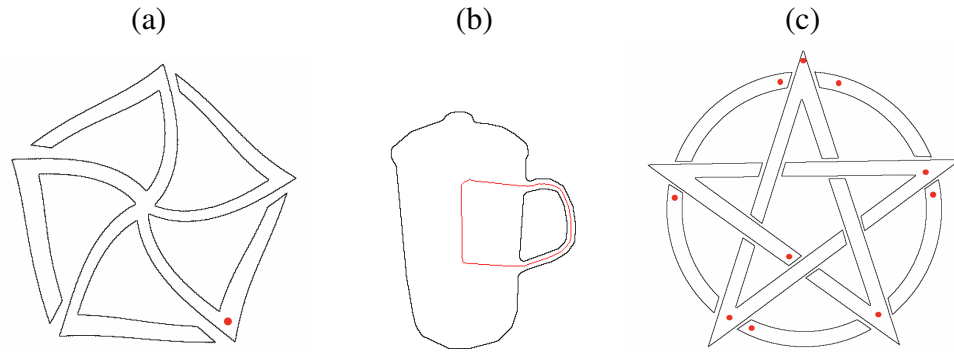


Figure 3.5: HJS with $\gamma < 1$ used as a homotopy classifier. (a) The skeleton is a single point, hence the shape is simply-connected. (b) The skeleton is homeomorphic to a circle, hence the shape is not simply-connected and has genus 1. (c) The skeleton consists of 10 points, hence the shape has ten simply-connected components. In (a) and (c), the identified skeleton points are emphasized by red disks for visualization.



Figure 3.6: HJS with $\gamma < 1$ used as a deficiency detector in binary shapes. (a) The given shape and identified non-trivial skeleton using $\gamma = 0.9$. (b) A hole on the boundary of the top-left petal. (c) A hole on the bottom-right pedal. (b) and (c) show the deficiencies inducing the non-trivial skeleton in (a). In all examples here, we keep $\gamma = 0.9$.

3.3.3 Shape Reconstruction from Medial Axis

Skeleton serves as a compact representation of the original shape. Combined with the distance function evaluated at the skeleton points, a medial axis allows shape reconstruction via taking a union of disks. In particular, from the set of skeleton points, $\text{Sk}(S_o)$, of the discrete shape S_o , our reconstructed discrete shape S_r is computed by

$$S_r = \bigcup_{(i',j') \in \text{Sk}(S_o)} \{(i,j) \in \Omega \cap \mathbb{N}^2 \mid \sqrt{(i-i')^2 + (j-j')^2} \leq D[i',j'] + \varepsilon\}. \quad (3.10)$$

Here, we introduce a *dilation parameter* $\varepsilon > 0$ to address the bias introduced by pixelization. We take $\varepsilon = 1.5$ as the default, which is justified later.

Figure 3.7 shows the reconstructed shapes from HJS using different values of γ . Recall from the previous discussion that, as $\gamma > 0$ increases, the identified skeleton consists of more branches representing details of the silhouette. Consistent with this feature characterization, from (b) to (d), as γ is increased from 0.9 to 20, more details of the original shape are recovered using the medial axis. Since $\gamma < 1$ for column (b), the reconstructed shape indicates existence of holes or simply connected components in the shapes in (a). In particular, the *sakura* in the first row is simply connected, the *knot* in the second row has 12 simply connected components, the *trophy* in the third row is of genus 6, and the *deer* in the fourth row contains many corrupted pixels (small holes) which are hard to observe at first glance. When $\gamma = 2.5$ (our default value), we recover most parts of the shapes in all cases. The results in column (d) are obtained using $\gamma = 20$, which recovers finer details of the original shapes compared to (c). For example, the sharp tips of the *petals*, the corners and T-junctions in the *knot*, the layers on the bottom of the *trophy*, and the elongated horns of the *deer*.

To quantify the reconstruction results, we compare the original discrete shape $S_o = \{(i,j) \in \Omega \cap \mathbb{N}^2 \mid I(i,j) = 0\}$ with the reconstructed shape S_r by considering the following

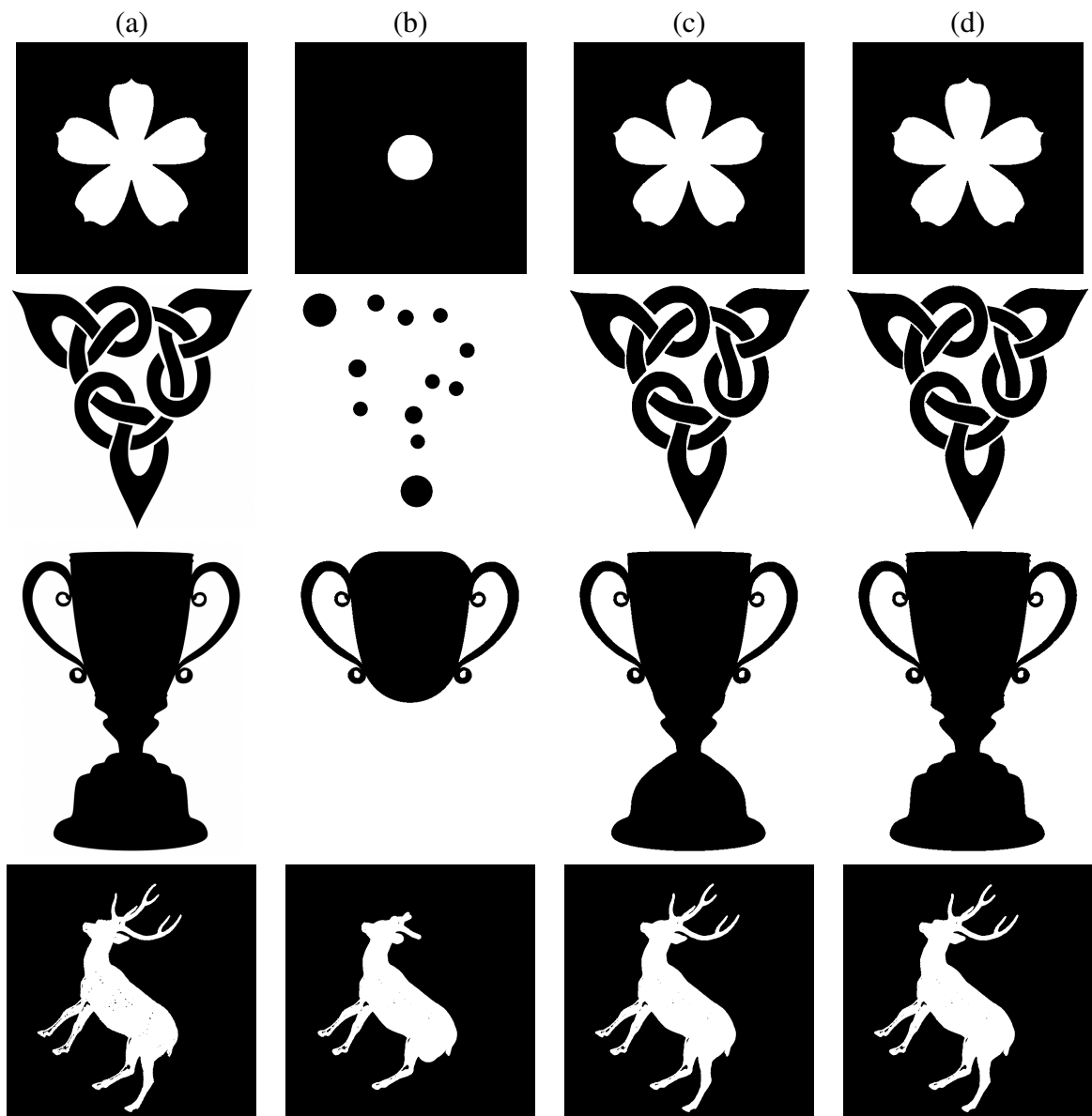


Figure 3.7: Shape reconstructed from the medial axis transform. (a) Original shapes. (b)-(d) The shapes reconstructed from the HJS using (b) $\gamma = 0.9$, (c) $\gamma = 2.5$, and (d) $\gamma = 20$. Here we fixed $\varepsilon = 1.5$.

three measures:

$$\begin{aligned}
\text{Jaccard Index [211]:} \quad & J = \frac{|S_o \cap S_r|}{|S_o \cup S_r|} . \\
\text{Dice Similarity Coefficient [212]:} \quad & DSC = \frac{2|S_o \cap S_r|}{|S_o| + |S_r|} . \\
\text{Bpn Bias Estimator [213]:} \quad & Bpn = \frac{|S_r \setminus S_o| - |S_o \setminus S_r|}{|S_o \cap S_r|} .
\end{aligned}$$

Here $|\cdot|$ denotes the cardinality of a set. Higher values of Jaccard index or Dice similarity coefficient indicate higher similarity between S_o and S_r . A positive (negative) Bpn bias estimator signifies that S_r is an over-(under-) coverage of S_o , and a zero value means no bias, i.e., the size of the over-coverage cancels out with the size of the under-coverage. These measures behave differently. For the Jaccard index, every element in the intersection is counted once, whereas for the Dice similarity coefficient, every element in the intersection is counted twice. In fact, the Jaccard index and the Dice similarity coefficient is related via $J = DSC / (2 - DSC)$, hence, compared to the Dice similarity coefficient, the Jaccard index is less sensitive to distinct objects, while more sensitive to similar objects. The Bpn bias estimator provides the additional information for avoiding over-coverage or under-coverage.

In Table 3.1, we report these measures for the results in Figure 3.7. In all cases, when we increase γ , the shape reconstructed from the medial axis becomes more similar to the original shape, which is characterized by the increasing J and DSC . Since we are using disks with radius enlarged by 1.5 pixels for reconstruction, Bpn 's change their signs from negative (under-coverage) to positive (over-coverage) as the endpoints of the skeleton approach the boundary of the original shapes, respectively. For comparison, we also include these measures when $\varepsilon = 0$. Using $\varepsilon = 1.5$ improves the reconstructions measured by higher J 's and DSC 's, and it produces less biases quantified by the smaller absolute values of Bpn 's. We notice that when $\gamma = 0$, Bpn 's always remain negative in our examples. Figure 3.8 provides a further investigation on the effects of varying ε as $\gamma = 2.5$ is fixed.

In both (a) and (b), the maximal rescaled values of J and DSC occur around $\varepsilon = 1$, but to acquire the minimal bias, larger values of ε are needed. This is due to the fact that we are using unions of finitely many disks to cover non-convex shapes, and slightly increasing ε allows balancing the over- and under-fitting. Hence, we recommend $\varepsilon = 1.5$, which leads to results with highly accurate shape reconstruction and low bias.

Table 3.1: Comparative measures of the reconstruction results in Figure 3.7 ($\varepsilon = 1.5$). Higher values of J and DSC indicate higher similarity between S_o and S_r . When Bpn is positive (negative), S_r is an over- (respectively under-) coverage for S_o , and when $Bpn = 0$, there is no bias. We report these measures when $\varepsilon = 0$ for comparison.

Sakura (1 st row)	$\gamma = 0.9$		$\gamma = 2.5$		$\gamma = 20$	
	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$
J	0.1267	0.1340	0.9566	0.9770	0.9715	0.9816
DSC	0.2247	0.2363	0.9778	0.9884	0.9855	0.9907
Bpn	-6.8943	-6.4625	-0.0454	0.0006	-0.0294	0.0187
Knot (2 nd row)	$\gamma = 0.9$		$\gamma = 2.5$		$\gamma = 20$	
	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$
J	0.1714	0.1911	0.9451	0.9719	0.9436	0.9479
DSC	0.2926	0.3209	0.9718	0.9858	0.9732	0.9831
Bpn	-4.8351	-4.2234	-0.0581	0.0272	-0.0550	0.0342
Trophy (3 rd row)	$\gamma = 0.9$		$\gamma = 2.5$		$\gamma = 20$	
	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$
J	0.5576	0.5729	0.9641	0.9787	0.9745	0.9832
DSC	0.7160	0.7285	0.9817	0.9892	0.9871	0.9915
Bpn	-0.7933	-0.7146	-0.0372	0.0043	-0.0261	0.0170
Deer (4 th row)	$\gamma = 0.9$		$\gamma = 2.5$		$\gamma = 20$	
	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$	$\varepsilon = 0$	$\varepsilon = 1.5$
J	0.8361	0.8691	0.9324	0.9736	0.9411	0.9662
DSC	0.9108	0.9300	0.9650	0.9866	0.9697	0.9828
Bpn	-0.1960	-0.1153	-0.0725	0.0174	-0.0626	0.0332

3.3.4 Performance of Distance Computation

As a natural extension of the Eikonal equation, the fast sweeping algorithm suits our purpose. In this section, we compare the fast sweeping algorithm with a brute-force method, and the celebrated [207], which is based on an optimization model efficiently solved using morphological operations.

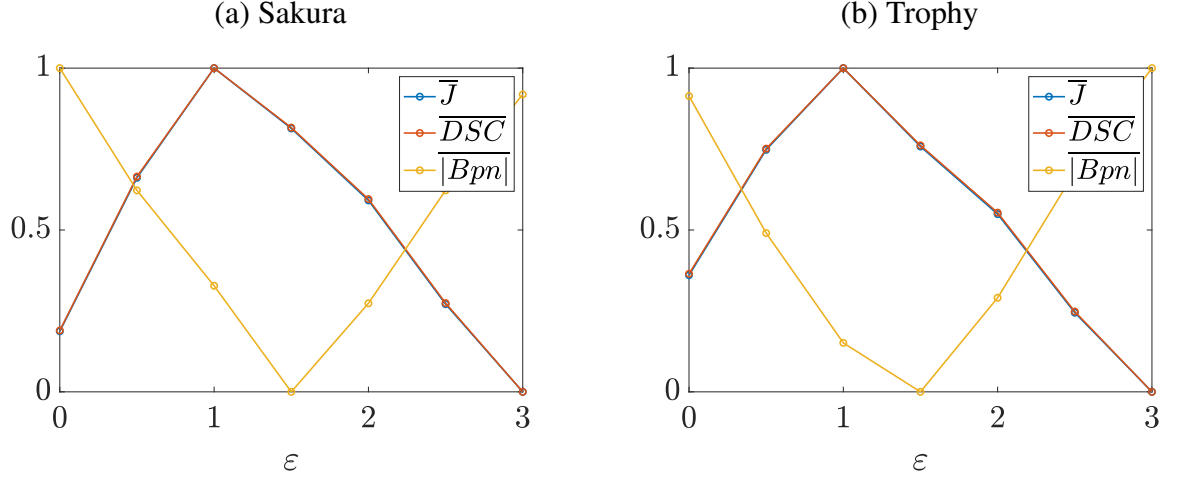


Figure 3.8: Effects of varying ε on the comparative measures. Here we plot the values of the rescaled measures, \bar{J} , \overline{DSC} and $|Bpn|$, against different values of ε , when HJS ($\gamma = 2.5$) is applied to the *sakura* in the first row and the *trophy* in the third row of Figure 3.7. Both plots indicate that using slightly dilated disks improves the reconstruction results.

On a discrete image domain, similarly to definition (Equation 3.1), one may define for each pixel $(i, j) \in \Omega \cap \mathbb{N}^2$, its distance to the boundary specified by \mathcal{B} by

$$D[i, j] = \min_{(i', j') \in \mathcal{B}} \sqrt{(i - i')^2 + (j - j')^2}. \quad (3.11)$$

The implementation is straightforward, and we present the pseudo-code in algorithm 4. If there are K boundary points, then the cost of the brute-force method is of the order of $O(KMN)$, and it can be reduced by focusing only on the interior points of the shape.

The Felzenszwalb-Huttenlocher (F-H) algorithm [207] views the distance transform as a minimum convolution of two functions. More specifically, the squared Euclidean distance to the boundary of the shape $A \subseteq \Omega$ is obtained via an optimization problem

$$D_{\text{F-H}}(x, y) = \min_{x', y'} ((x - x')^2 + (y - y')^2 + \chi_{\partial A}(x', y')) \quad (3.12)$$

where $\chi_{\partial A}(x', y') = 0$ if $(x', y') \in \partial A$ and $+\infty$ otherwise. The discretized version of Equation 3.12 is efficiently solved via two main steps in each dimension. First, compute the

Input: I a binary image which takes value 0 for points inside the shape, and 1 for background points

Compute the set of boundary points \mathcal{B} .

Assign $D[i', j'] = 0$ for every boundary point $(i', j') \in \mathcal{B}$.

```

for  $(i, j) \in \Omega \setminus \mathcal{B}$  with  $I[i, j] = 0$  do
    Set  $D[i, j] = M^2 + N^2$ .
    for  $(i', j') \in \mathcal{B}$  do
        if  $(i - i')^2 + (j - j')^2 < D[i, j]$  then
             $D[i, j] = (i - i')^2 + (j - j')^2$ .
        end
    end
    Update  $D[i, j] \leftarrow \sqrt{D[i, j]}$ .
end

```

Output: Distance map D for the interior points of the shape in I .

Algorithm 4: Distance Computation: A Brute-force Method

lower envelope of parabolas induced by each grid point's squared Euclidean distance function; at this stage, the minimum convolution is employed. Second, the distance values at grid points are filled in by comparing them with the computed lower envelope. The total computational cost is $O(2MN)$.

We applied these three methods to the shape of a cat in Figure 3.9 (a). The distance transform computed using the brute-force method is shown in (b), the distance transform obtained using the fast sweeping algorithm is in (c), and the distance transform by the F-H algorithm is in (d). Noticeably, the computation of both fast sweeping algorithm (141.22 ms) and F-H algorithm (23.16 ms) are much faster than the brute-force method (3424.08 ms), yet all these methods produce similar results. The skeletons computed based on the distance transform of these methods are respectively displayed in (e), (f), and (g). Notice that compared to (e) and (g), the skeleton in (f) obtained based on the fast-sweeping has fewer short branches and more robust against small-scale fluctuations on the shape's boundary. This was expected since the fast sweeping algorithm's distance transform approximates a diffusive solution of the Eikonal equation. Hence, (c) can be regarded as a slightly smoothed version of the exact distance transform.

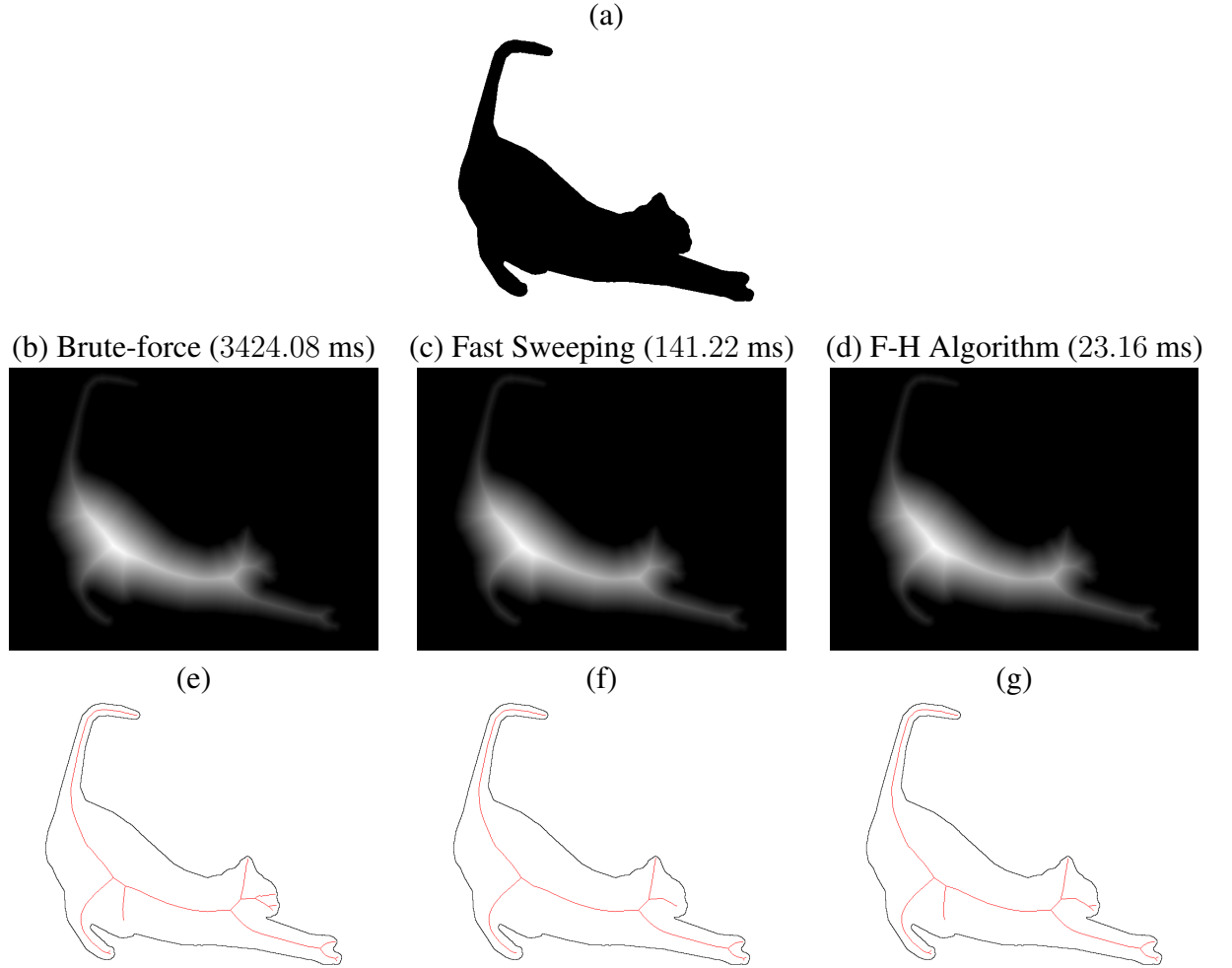


Figure 3.9: (a) Binary image (537×700): a cat silhouette. The distance function is computed by (b) a brute-force method (algorithm 4), (c) the fast sweeping algorithm (algorithm 2), and (d) the F-H algorithm [207]. Brighter pixels indicates further distance from the contour. The F-H algorithm is the fastest, then the fast-sweeping, and the brute-force is the slowest. (e) shows the skeleton computed based on the distance transform in (b); (f) shows the skeleton computed based on the distance transform in (c); and (g) shows the skeleton computed from (d). In all cases, we fixed $\gamma = 2.5$.

3.4 Contour-based Representation – Silhouette Vectorization

A silhouette appears as one of the connected components of an upper or lower set of the image. More generally, the study of shape promoted by Mathematical Morphology [214] defines 2D shapes as any such connected component. Silhouettes are essential for the human perception of shapes, and the distribution of corners along their outlines are closely linked to the psychophysical models of the visual system [215]. They are also fundamental to image representation [216] where an image can be decomposed into a tree of connected shapes ordered by inclusion.

There are various ways to characterize silhouettes. As proved in [217], if a closed subset of the plane has a finite perimeter, then it can be described by its essential boundary, which is a countable set of Jordan curves with finite length. From digital images, upper-level sets can be extracted by mere thresholding, in which case they are a finite union of pixels, bounded by a finite number of Jordan curves made of vertical and horizontal segments. Using a parametric interpolation such as the bilinear, one can extract the boundary of a level set as a union of pieces of hyperbolae [218]. In a founding work, Montanari [219] introduced a polygonal approximation of outlines of rasterized silhouettes. After the discrete boundary is traced, the polygonal vertices' sub-pixel locations are determined by minimizing global length energy with an L^∞ loss to the initial outline.

In modern computer graphics, describing a silhouette's boundary as a union of primitive components, such as line segments, circular arcs, and Bézier curves is more popular for their real-time rendering [220]. The scalable feature of vectorization is crucial for 2D shapes such as logos or fonts, which require printing in different sizes. The geometric features captured by the vectorization are also important in feature identification [221], remote sensing [222], and others applications [223, 224, 225].

Common silhouette vectorization methods [226, 227, 228, 229, 230, 223, 231, 232] consist of two steps: identification of control points and approximation of curves connect-

ing the control points. Ramer [227] proposed an iterative splitting scheme for identifying a set of control points on a polygonal line \mathcal{C} such that the Bézier polygon $\hat{\mathcal{C}}$ defined by these vertices approximates \mathcal{C} in L^∞ norm. The Hausdorff distance between $\hat{\mathcal{C}}$ and \mathcal{C} is constrained to stay below a predefined threshold, and the number of control points is sub-optimal. More recently, Safraz [232] proposed an outline vectorization algorithm that splits the outline at corners which are identified without computing curvatures [233], then new control points are introduced to improve curve fitting.

The control points produced by many methods are near curvature extrema of the outline, but this may happen by algorithmic convergence rather than by an explicit design. It is well-known that the direct computation of curvature is not reliable [234]. We shall use the affine scale-space [235] on curves to detect corners. Our method is related to [236, 237] where the authors use the Affine Morphological Scale Space (AMSS) [238] to define a morphological *corneriness measure* based on the expected evolution of an ideal corner through AMSS. The authors use a closed-form expression for the evolution of a perfect corner under AMSS as a “traveling wave”. Their method applies to raster digital images using a finite difference scheme for AMSS and is expanded to analyze multiple junctions. Our setup here will use Moisan’s geometric curve evolution scheme [239], which processes a Jordan curve and ensures a natural sub-pixel positioning of the curvature extrema. Our experience being that many tips in shapes have a more complex geometry than a straight corner, we do not use a closed-form expression for their evolution. We rely on the original scale-space paradigm, that essential features can be detected at coarse scales and then backtracked in scale space to their initial position. While the name of scale-space [240] is associated with this method, the invention of the method for shape analysis is much anterior and goes back to the Japanese school of shape analysis [241], in particular, the founding works of T. Iijima [242] on character recognition. The methods mentioned above reflect the challenges of estimating the outline’s curvature on shapes extracted from raster images.

3.5 Outline of the Affine-scale Space Vectorization Procedure

We introduce a novel vectorization approach [243] fundamentally based on mathematical advances for their stability and sub-pixel accuracy. The proposed method has three main steps that work together to find geometrically meaningful control points: it first identifies (i) curvature extrema of the outline computed at the sub-pixel level, by (ii) backpropagating control points detected as curvature extrema at coarser scale in the affine scale-space, then (iii) computing piecewise least-square cubic Bézier joining these control points while fitting the smoothed outline with a predefined accuracy. We describe the outline of the method here, and leave the details in later sections.

On a rectangular domain $\Omega = [0, H] \times [0, W] \subset \mathbb{R}^2$ with $H > 0$ and $W > 0$, a *silhouette* is a compact subset $\mathcal{S} \subset \Omega$ whose topological boundary $\partial\mathcal{S}$, the *outline*, is a piecewise smooth curve. Suppose \mathcal{S} is shown in a *raster binary image* $I : \Omega \cap \mathbb{N}^2 \rightarrow \{0, 255\}$, that is, the set of black pixels

$$\bar{\mathcal{S}} = \{(i, j) \in \Omega \cap \mathbb{N}^2 \mid I(i, j) = 0\}$$

approximates \mathcal{S} . We assume that $\mathcal{S} \cap \partial\Omega = \emptyset$. The main objective of this paper is to find a cubic Bézier polygon close to $\partial\mathcal{S}$ in the Hausdorff distance such that the vertices are geometrically meaningful. As a result, the proposed algorithm takes any binary raster image and converts it to an SVG file with compact size.

Figure 3.10 shows the overview of the proposed method. From the input image (a), which is a pixelized raster image, bilinear outlines are computed in (c), and affine scale-space is used to find the control points in (f). By cubic Bézier polygon, the vectorized result is presented in (g). Images (a) and (h) show the zoom-in of one of the corners, which illustrates a sharp representation of the given raster image in the vectorized form. Every step is designed to fully explore mathematically and geometrically meaningful features of the silhouette, utilizing the techniques which promote affine invariance. This approximation

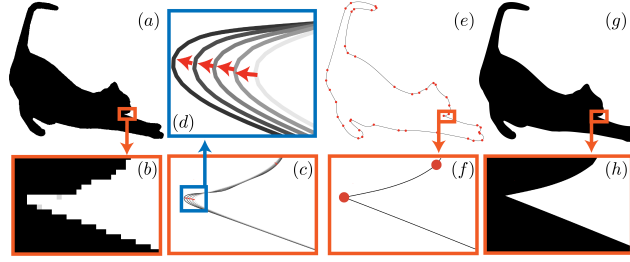


Figure 3.10: A flowchart of the proposed method. (a) A given raster image of a *cat*'s silhouette. (b) Zoom-in of (a). (c) Extracted bilinear outline of (a). (d) Inversely tracing the curvature extrema along the affine shortening flow. (e) The vectorized outline of (a) with control points marked as red dots. (f) Zoom-in of (e). (g) Vectorized result of silhouette (a) by the proposed method. (h) Zoom-in of (g). Notice the improvement from the given raster image (a) to the proposed method's result in (g), as well as the zoom of (b) and (h).

guarantees subpixel accuracy in reconstruction.

There are mainly three steps in our proposed vectorization method.

- **Computation of sub-pixel curvature** (section 3.6): In the first step, we find the outline curvature extrema by computing curvatures of the bilinear level lines at arbitrary resolutions [244]. Compared to the conventional finite difference approach, this method yields a more accurate and visually conformal evaluation of the curvatures. It is essential to smooth the level lines first when working with raster images [219]. The affine shortening [235] preserves affine invariants of the outline while effectively reducing the staircase effects [245]. To avoid grid-dependence, we use a fully consistent geometric scheme [239] for the affine shortening partial differential equation.

- **Identification of control points** (section 3.7) Secondly, we propose to use Witkin's scale space strategy [240] adapted to the affine scale space to identify the geometrically meaningful control points. To sort out the "real curvature extrema" in the initial fine-scale curve, which is noisy and aliased, the affine scale space is applied to the initial curve. The initial curve has many curvature extrema, but most of them are irrelevant to the general shape. Using the scale-space approach, the (fewer) curvature extrema detected at a coarser scale are traced back to the initial fine-scale outline. Since the curve's normal direction at any scale is well-defined, we identify and trace back the curvature extrema from the coarser

scale to the finer scale by reversing the affine scale space in the normal direction. Note that Witkin's scale space strategy was used to detect the signal's or image's gradient's relevant extrema, namely those that persist after applying the heat equation to the initial datum. In this paper, the scale-space strategy is used, not to detect image edges, but to identify relevant curvature extrema.

- **Refinement of control points** (section 3.8). In the final step of our vectorization, we fit piecewise cubic Bézier curves to the outline's segments joining the control points. Similar to [229], we set a threshold for the global error, and our algorithm adaptively inserts new control points to guarantee that accuracy. We also consider the degenerate cases where no curvature extrema are identified, e.g., a circle or a smoothly varying planar curve in a high-resolution image.

In the following sections, we present the details.

3.6 Sub-pixel Curvature Extrema Localization

Our control points are largely derived from the outline's curvature extrema. To compute sub-pixel curvatures, we start with the bilinear interpolation [244] $u : \Omega \rightarrow [0, 255]$ such that

$$u(i + 1/2, j + 1/2) = I(i, j) , \quad (i, j) \in \Omega \cap \mathbb{N}^2 .$$

Here $\lfloor r \rfloor$ is the floor function giving the greatest integer smaller than the real number r . For any $\lambda \in (0, 255)$, the level line of u corresponding to λ is defined as $\mathcal{C}_\lambda = \{(x, y) \in \Omega \mid u(x, y) = \lambda\}$. It approximates the discrete outline as a piecewise C^2 Jordan curve except for at finitely many points, e.g., saddle points [246]. Fixing any non-integer $\lambda^* \in (0, 255)$, \mathcal{C}_{λ^*} is either piecewise linear (horizontal or vertical) or a part of a hyperbola whose asymptotes are adjacent edges of a single pixel.

Due to pixelization, \mathcal{C}_{λ^*} shows strong staircase effects [245], which causes unstable

curvature computation. Such oscillatory behavior is effectively reduced by the affine shortening flow [245, 235] by evolving the noisy curve \mathcal{C} by the following time-dependent PDE

$$\frac{\partial \mathcal{C}(s, t)}{\partial t} = \kappa^{1/3}(s, t) \mathbf{N}(s, t), \quad \mathcal{C}(s, 0) = \mathcal{C}(s), \quad t \geq 0 \quad (3.13)$$

till some short time $T_0 \geq 0$. Here each curve $\mathcal{C}(\cdot, t)$ is arc-length parametrized $s \in [0, \text{Length}(\mathcal{C}(\cdot, t))]$ for any t , κ denotes the signed scalar curvature, and \mathbf{N} is the inward unit normal at $\mathcal{C}(s, t)$. This process is independent from the viewpoint on the shape [247, 248].

Denoting the smooth bilinear outline by Γ_{λ^*} , at any vertex $P \in \Gamma_{\lambda^*}$, its unit normal direction $N(P)$ is computed by central difference, and its curvature $\kappa(P)$ is approximated by the curvature of the circumcircle that passes through three consecutive vertices on Γ_{λ^*} [249]. The discrete curvature values can be obtained at arbitrary resolution based on the sampling frequency applied to the bilinear outline \mathcal{C}_{λ^*} .

3.7 Affine Scale-space Control Points Identification

The curvature extrema computed above capture the abrupt geometrical changes in the smooth bilinear outline, but they are sensitive to noise. We propose to filter the control points by incorporating varying geometric scale of the outline based on the affine scale-space.

3.7.1 Backward Tracing via Inverse Affine Shortening Flow

The set of solutions of Equation 3.13 at different time $t \geq 0$, i.e., $\{\mathcal{C}(\cdot, t)\}_{t \geq 0}$ defines an affine scale-space [235], and the non-negative parameter t is called *scale*. This parametric space satisfies the causality:

Proposition 3.7.1. [235] *In the affine invariant scale-space of a planar curve, the number of extrema of Euclidean curvature is a nonincreasing function of time.*

In other words, every curvature extremum on the curve at a coarser scale. i.e., at larger t , is the continuation of at least one of the extrema at a finer scale, i.e., at smaller t . The lack of one-to-one correspondence is due to the possibility of multiple extrema (e.g., two maxima and one minimum) merging to a single one during the evolution. By tracing curvature extrema from the coarser scales to the finer scales, the resulting extrema are robust to noise and help capture prominent corners.

We define the control points as the curvature extrema on Γ_{λ^*} which persist across different scales in its affine scale-space. Given a sequence of discrete scales $t_0 = 0 < t_1 < \dots < t_K$ for some positive integer K , we obtain the curve $\mathcal{C}(\cdot, t_n)$ at scale t_n by the affine shortening flow Equation 3.13 for $n = 0, 1, \dots, K$. For any $1 \leq n \leq K$, the affine shortening flow Equation 3.13 is approximated as

$$\frac{\mathcal{C}(s, t_n) - \mathcal{C}(s, t_{n-1})}{t_n - t_{n-1}} = (\kappa^n(s))^{1/3} \mathbf{N}^n(s) + \mathbf{r}(s) , \quad (3.14)$$

where κ^n and \mathbf{N}^n denote the curvature and normal at the scale t_n , and \mathbf{r} is a remainder such that $\|\mathbf{r}(s)\| = O(t_n - t_{n-1})$. Rearranging Equation 3.14 gives

$$\begin{aligned} \mathcal{C}(s, t_{n-1}) &= \mathcal{C}(s, t_n) - (t_n - t_{n-1})(\kappa^n(s))^{1/3} \mathbf{N}^n(s) + \\ &\quad (t_n - t_{n-1})\mathbf{r}(s) . \end{aligned}$$

This expression shows that, if $t_n - t_{n-1}$ is sufficiently small, by following the opposite direction of the affine shortening flow at $\mathcal{C}(s, t_n)$, that is,

$$-\text{sign}(\kappa^n(s)) \mathbf{N}^n(s) ,$$

we can find $\mathcal{C}(s, t_{n-1})$ nearby. Here $\text{sign}(r)$ denotes the sign function which gives $+1$ if $r > 0$, -1 if $r < 0$ and 0 if $r = 0$. This gives a well-defined map from the curve at a coarser scale t_n to a finer scale t_{n-1} via the inverse affine shortening flow.

Starting from K , for any curvature extremum X^K on $\mathcal{C}_K = \mathcal{C}(\cdot, t_K)$, we set up the following constrained optimization problem to find a curvature extremum X^{K-1} on \mathcal{C}_{K-1} at scale t_{K-1} :

$$\begin{aligned} & \max_{X \in \mathcal{C}_{K-1}} \frac{\langle X - X^K, -\text{sign}(\kappa^K) \mathbf{N}^K \rangle}{\|X - X^K\|} \\ & \text{s.t.} \begin{cases} \frac{\langle X - X^K, -\text{sign}(\kappa^n) \mathbf{N}^n \rangle}{\|X - X^K\|} > \alpha, \\ \|X - X^K\| < D, \text{ and} \\ X \text{ is a curvature extremum on } \mathcal{C}_{K-1}, \end{cases} \end{aligned} \quad (3.15)$$

where $D > 0$ is a positive parameter that controls the closeness between X and X^K , and α enforces that the direction of $X - X^K$ is similar to that of the inverse affine shortening flow. The problem Equation 3.15 looks for the curvature extremum on \mathcal{C}_{K-1} in the D -neighborhood of X^K that is the nearest to the line passing X^K in the direction of the inverse affine shortening flow. When D and α are properly chosen, if Equation 3.15 has one solution, we define it to be X^{K-1} . If Equation 3.15 has multiple solutions, we choose the one with the shortest distance from X^K to be X^{K-1} . In case multiple solutions are having the same shortest distance from X^K , we arbitrarily select one to be X^{K-1} . In practice, if Equation 3.15 has a solution, it is almost always unique.

We repeat Equation 3.15 for decreasing $K-1, K-2, \dots, 0$. Either the solutions always exist until the scale t_0 , or there exists some $m \geq 1$, such that Equation 3.15 at t_m does not have any solution. In the first case, we call X^K a *complete* point, and in the second case, we call it *incomplete*. For each curvature extremum X^K on \mathcal{C}_K , we construct a sequence of points $\mathcal{L}(X^K)$ that contains the solutions of Equation 3.15 for $K, K-1, K-2, \dots$, starting at X^K in a scale-decreasing order. If X^K is complete, then $\mathcal{L}(X^K)$ has exactly $K+1$ elements, and we call the sequence complete; otherwise, the size of $\mathcal{L}(X^K)$ is strictly smaller than $K+1$, and we call the sequence incomplete.

We define the last elements of the complete sequences as the candidate control points,

and denote them as $\{O_i(t_K)\}_{i=1}^{M(t_K)}$. These points are ordered following the orientation of Γ_{λ^*} . The parameter t_K in the parenthesis indicates that the candidate control points are associated with the curvature extrema identified at the scale t_K . When the scale t_K is fixed, we simply write $\{O_i\}_{i=1}^M$.

This inverse affine scale-space approach prioritizes the curvature extrema, which persist across different affine shortening flow scales. This step is essential in keeping geometrically meaningful control points and reducing the total number of control points.

3.7.2 Degenerate Case

When the underlying silhouette is a disk or has a smoothly varying boundary, provided that the image has a sufficiently high resolution, there may not be any candidate control points identified on Γ_{λ^*} associated with the curvature extrema at scale t_K . We call it a *degenerate case*.

If S is a disk, the vectorization only requires its center and radius. We use the isoperimetric inequality to determine if Γ_{λ^*} represents a circle: for any closed plane curve with area A and perimeter L , we have $4\pi A \leq L^2$ and the equality holds if and only if the curve is a circle. In practice, we decide that Γ_{λ^*} is a circle only if the corresponding ratio $1 - 4\pi A/L^2$ is sufficiently small. By this criterion, if Γ_{λ^*} is classified as a circle, its center and radius are easily computed by arbitrarily three distinct points on Γ_{λ^*} . For numerical stability, we take three outline points that are equidistant from each other. Otherwise, we insert a pair of most distant points on Γ_{λ^*} to be the candidate control points. An efficient approach for finding these points is to combine a convex hull algorithm, e.g., the monotone chain method [250], which takes $O(N \log N)$ time, with the rotating calipers [251], which takes $O(N)$ time. Here N is the number of vertices of the polygonal line Γ_{λ^*} .

3.8 Adaptive Cubic Bézier Polygon Approximation

After the control points are identified from the affine scale-space, $H := \{O_i\}_i^M$, we adjust H by deleting non-salient sub-pixel curvature extrema and inserting new control points for guaranteeing a predefined accuracy. This adaptive approach yields a cubic Bézier polygon $\mathcal{B}(H)$ whose vertices are points in H and edges are cubic Bézier curves computed by least-square fittings.

3.8.1 Bézier Fitting with Chord-length Parametrization

A cubic Bézier curve is specified by four points B_0, B_1, B_2 , and B_3 . Its parametric form is

$$B(s) = (1-s)^3 B_0 + 3(1-s)^2 s B_1 + 3(1-s)s^2 B_2 + s^3 B_3,$$

for $s \in [0, 1]$. Specifically, it has the following properties: (i) B_0 and B_3 are the two endpoints for $B(s)$; and (ii) $B_1 - B_0$ is the right tangent of $B(s)$ at B_0 , and $B_2 - B_3$ is the left tangent at B_3 . To approximate a polygonal line segment $\Sigma = \{P_0, P_1, \dots, P_N\}$, we find a cubic Bézier curve that is determined by $B_0 = P_0$, B_1 , B_2 , and $B_3 = P_N$ such that the squared fitting error

$$\begin{aligned} \tilde{S} = \sum_{i=0}^N & \left(P_i - ((1 - \tilde{s}_i)^3 B_0 + 3(1 - \tilde{s}_i)^2 \tilde{s}_i B_1 + \right. \\ & \left. 3(1 - \tilde{s}_i) \tilde{s}_i^2 B_2 + \tilde{s}_i^3 B_3) \right)^2 \end{aligned} \quad (3.16)$$

is minimized. Here $\tilde{s}_i = (\sum_{k=1}^i \|P_k - P_{k-1}\|) / (\sum_{k=1}^N \|P_k - P_{k-1}\|)$ is the chord-length parameter for P_i with $i = 0, 1, \dots, N$. We note that Equation 3.16 is used to initialize an iterative algorithm in [252] for a more accurate Bézier fitting. The benefit of this approximating setup is that we have closed-form formulae [229] for the minimizing B_j , $j = 1, 2$,

Hence we gain computational efficiency.

3.8.2 Control Point Refinement: Deletion of Sub-pixel Extrema

Recall that the candidate control points $H = \{O_i\}_{i=1}^M$ in section 3.7 are curvature extrema at sub-pixel level. Hence they may not reflect salient corners of the silhouette. To remove spurious sub-pixel extrema from H , we compare the left tangent and right tangent at each candidate control point.

We take advantage of the second property of cubic Bézier curves mentioned in subsection 3.8.1. For $i = 1, \dots, M$, we fit a cubic Bézier to the polygonal line segment whose set of vertices is

$$\{O_i = P_{j(i)}, P_{j(i)+1}, \dots, P_{j(i+1)} = O_{i+1}\},$$

where we take $O_{M+1} = O_1$, and obtain the estimated defining points $B_{i,1}$ and $B_{i,2}$ for the Bézier curve. The left and right tangent at O_i are computed as

$$T_i^- = B_{i-1,2} - O_i, \quad T_i^+ = B_{i,1} - O_i, \quad (3.17)$$

respectively, where $B_{-1,2} = B_{M,2}$. These tangent vectors are associated with all the points between neighboring candidate control points. Therefore, the angle formed by T_i^- and T_i^+ measures the sharpness of Γ_{λ^*} at O_i from a more global perspective. We delete O_i from the set of candidate control points H if

$$\frac{\langle T_i^+, T_i^- \rangle}{\|T_i^+\| \|T_i^-\|} + 1 < \varepsilon,$$

for some small $\varepsilon > 0$. It is equivalent to the condition that the angle between T_i^+ and T_i^- is close to π . The set H is updated with the remaining control points.

When all the candidate control points $\{O_i\}_{i=1}^M$ are removed after this procedure, we

encounter a degenerate case. If the underlying outline is a circle, we compute the center and radius; if it is not, we take the most distant pair of outline points to update H .

3.8.3 Control Point Refinement: Insertion for Accuracy

The candidate control points in H split the outline Γ_{λ^*} into polygonal line segments, each of which is approximated by a cubic Bézier using least square fitting as described in subsection 3.8.1. We obtain a Bézier polygon that approximates Γ_{λ^*} , denoted by $\mathcal{B}(H)$. A natural measure for the error of approximating Γ_{λ^*} using the Bézier polygon $\mathcal{B}(H)$ is

$$e = \max_{P_i \in \Gamma_{\lambda^*}} \text{dist}(P_i, \mathcal{B}(H)) , \quad (3.18)$$

where $\text{dist}(P_i, \mathcal{B}(H)) = \inf_{P \in \mathcal{B}(H)} \|P_i - P\|$ is the distance from P_i to the curve $\mathcal{B}(H)$. It is desirable that the user can specify the threshold for the error, $\tau_e > 0$. To guarantee that $e \leq \tau_e$, we apply the splitting strategy [227] which inserts $P_{\text{new}} \in \Gamma_{\lambda^*}$ to H as a new control point if

$$\text{dist}(P_{\text{new}}, \mathcal{B}(H)) > \tau_e , \quad (3.19)$$

and among those points on Γ_{λ^*} satisfying Equation 3.19, the distance from P_{new} to $\mathcal{B}(H)$ is the largest. After the insertion, we fit Γ_{λ^*} using a Bézier polygon based on the new set of control points in H . If the error of the newly fitted Bézier polygon is still greater than τ_e , we insert another point based on the same criterion. This series of insertions terminates once the condition $e \leq \tau_e$ is met.

Finally, $\mathcal{B}(H)$ with the updated set of control points H gives a Bézier polygon that approximates the outline $\partial\mathcal{S}$. With its interior filled with black, we obtain the vectorized silhouette for \mathcal{S} from the raster image I .

Remark 3.8.1. *For a further reduction on the size of H , we may consider an optional step to merge neighboring Bézier cubics if the union of the underlying polygonal line segments*

can be approximated by a single Bézier cubic via Equation 3.16 with an error below τ_e . We can regard the insertion in subsection 3.8.3 as controlling the data fidelity, and the simplification described here as minimizing the complexity of an estimator. Alternatively iterating these procedures provides a numerical scheme for a constrained optimizing problem $\min_{H \subseteq \Gamma_{\lambda^*}} |H|$ under the constraint that $\max_{P_i \in \Gamma_{\lambda^*}} \text{dist}(P_i, \mathcal{B}(H)) \leq \tau_e$, where $|H|$ denotes the number of elements in H . For any $\tau_e \geq 0$, it always has a solution, yet the uniqueness largely depends on the geometric structure of Γ_{λ^*} .

3.9 Numerical Experiments on Silhouette Vectorization

We present a comprehensive set of numerical experiments to evaluate and compare the proposed algorithm's performance for the criteria of compression, accuracy, stability, complexity, and repeatability. We start with a description of the data and implementation details.

3.9.1 Data Preparation and Parameter Settings

After obtaining the SVGs from [253], we rasterized them as PNG images, which were used as inputs in the following experiments. The inputs were either binary or gray-scale. We extracted the level line for $\lambda^* = 127.5$ to approximate the outlines throughout the experiments.

To solve Equation 3.13, we apply the fully consistent geometric scheme [239] which is independent of grid discretization. Consequently, the scale parameter t is conveniently replaced by a chord-area parameter σ . The scale T_0 for the initial smoothing (section 3.6) required for curvature computation thus corresponds to some *smoothness parameter* σ_0 . The computed discrete curvatures are filtered by moving average with periodic boundary condition to suppress the noise. A curvature extremum is identified only if it has absolute value greater than its neighbors and above 0.001. For the parameters in Equation 3.15, we fixed $D = 10$ and $\alpha = 0.9$. During the inverse tracing (subsection 3.7.1), $K = 4$,

and since the sequence of scales $\{t_k\}_{k=1}^K$ can be replaced by chord-area parameters, the curvature extrema were traced for scales corresponding to chord-areas $k\Delta\sigma$, $k = 1, 2, 3, 4$ respectively, where $\Delta\sigma = 0.5$. The threshold for the degenerate case (subsection 3.7.2) is set to be 0.005.

By default, we set the error threshold $\tau_e = 1$, so that the vectorized outline was guaranteed to have sub-pixel accuracy; and the smoothness parameter $\sigma_0 = 1$. Table B.1 collectively displays the silhouettes used in the following experiments.

3.9.2 General Performance

We present some results of our proposed algorithm in Figure 3.11. In (a), we have a silhouette of a *cat*. It has a single outline curve that contains multiple sharp corners on the tail, near the neck, and around the paws. These features provide informative visual cues for silhouette recognition, and our algorithm identifies them as control points for the silhouette vectorization shown as the red dots in (b). The outline of a *butterfly* in (c) has multiple connected components. In addition to the control points corresponding to corners, we observe in (d) some others on smooth segments of the outline. They are inserted during the refinement step of our algorithm, where a single Bézier cubic is inadequate to guarantee the accuracy specified by the error threshold $\tau_e = 1$. In (e), we show a tessellation of words, and (f) presents the vectorized result. The input is a PNG image of dimension 1934×1332 and takes 346 KB in the storage. In contrast, its silhouette vectorization, saved as an SVG file, has 2683 control points and takes 68 KB if the coordinates are stored in float. In this example, our algorithm provides a compression ratio of about 80.35%. The total computational time for this case only takes 0.83 seconds. We also compared the resulted size with the lossy compression format JPG to show that the proposed method has a superior compression ratio. More statistics are summarized in Table 3.2.

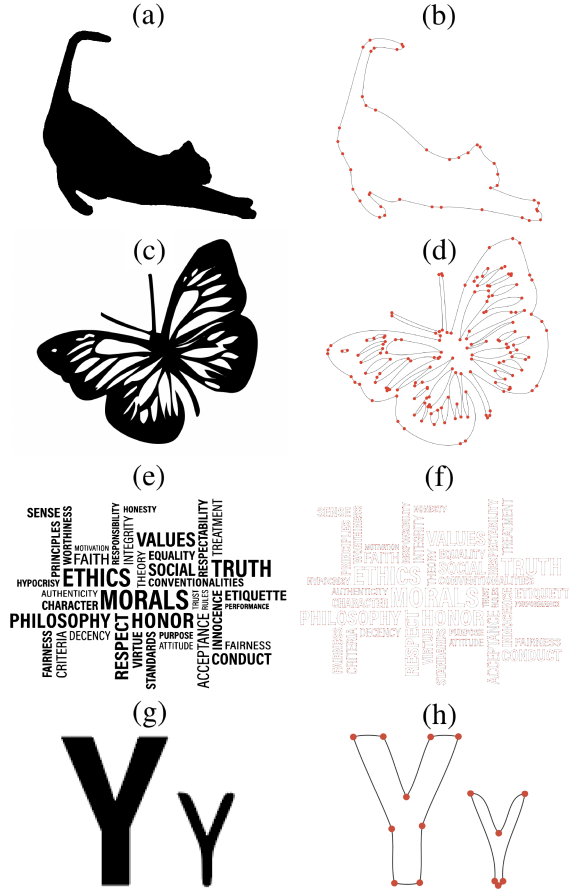


Figure 3.11: General performance. (a) *Cat* and (b) its vectorized outline (42 control points). (c) *Butterfly* and (d) its vectorized outline (158 control points). (e) Text design and its vectorized outline (2683 control points). Each red dot signifies the location of a control point. (g) Two letters exerted from (e) scaled up with the same magnitude. (h) Zoom-in of the vectorization (f) on the two letters in (g).

Table 3.2: Performance of the proposed method applied to examples in Figure 3.11. The compression ratios are displayed for PNG and JPG, respectively. *We note that the PNG image in (a) has a single channel, thus converting it to JPG increases the size.

Shape	*(a)	(c)	(e)
PNG Size	5 KB	178 KB	346 KB
JPG Size	11 KB*	35 KB	155 KB
Result Size	2 KB	5 KB	68 KB
Ratio (PNG)	60%	97.19%	80.35%
Ratio (JPG)	81.82%	85.71%	56.13%
Proc. Time	0.10 Sec.	0.15 Sec.	0.83 Sec.

3.9.3 Tests on Degenerate Cases

An important feature of our algorithm is that it has flexibility for degenerate cases, where the silhouette does not have identifiable curvature extrema on its outline, e.g., Figure 3.12 (a). Once our algorithm classifies the outline as a circle, instead of fitting Bézier cubics, it directly approximates the center and radius of the circle and draws a perfect circle. See Figure 3.12 (b). We note that Bézier curves cannot perfectly fit a circle [254], and it requires more than 6 distinct control points, whereas we only need one control point for the center and one scalar for the radius.

Figure 3.12 (c) shows another degenerate case. It consists of a rectangle in the middle and two half disks attached on its opposite sides, whose diameters are equal to the rectangle's height. This particular silhouette has no strict curvature extrema on its outline. By computation, its area is 172644 and perimeter is 1742.07; since $4\pi \text{Area}/\text{Perimeter}^2 = 4 \times \pi \times 172644/(1742.07)^2 = 0.7149 < 1$, the outline is not a circle. Hence the algorithm inserts a pair of most distant points on the outline, the left-most and the right-most points in this case, and conducts the Bézier fitting routine for the non-degenerate cases.

The design of this special procedure for degenerate cases is important for two reasons. First, it makes the algorithm adaptive to image resolutions. If we reduce the resolution of (c) from 774×320 to 144×58 , whose magnified version is shown in (e), due to strong pixelization, all the control points are identified as local curvature extrema. (f) shows the magnified vectorization of the low-resolution image. Second, it improves the compression ratio. To fit a circle using a Bézier polygon requires at least two pieces of cubics; hence we need to store the coordinates of at least 6 points. With our algorithm, only the center's coordinate and the value of the radius are required, which saves the space for 9 float or int type data. Figure 3.12 (g) shows mixture of degenerate and non-degenerate outline curves. The vectorization in (h) shows that the circles are represented as perfect circles, and the others are represented as Bézier polygons.

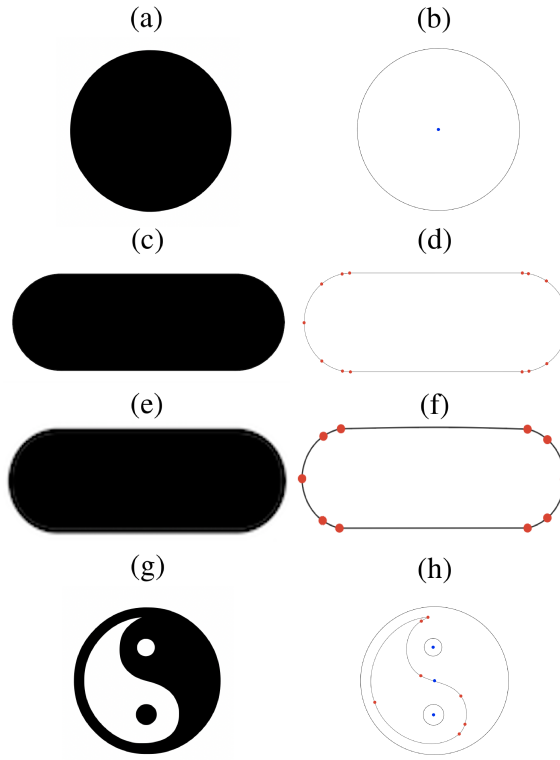


Figure 3.12: Degenerate cases. In (a) and (c), no candidate control points were identified. Our algorithm handles such situations by checking if the outline is a circle. If it is, e.g. (a), the center and radius are computed, and a circle is drawn without Bézier fitting; hence, there is no control point (red dots) on the vectorized outline (b). The blue dot indicates the center of the circle. If it is not a circle, e.g., (c), a pair of most distant points are inserted to initiate the Bézier fitting, such as in (d). (e) shows the low-resolution version of (c), and (f) displays its vectorization. When the resolution is low, all the control points are identified curvature extrema. In (g), three of the outline curves are identified as circles, and the others are fitted by Bézier polygons. (h) shows the vectorized result.

3.9.4 Effect of the Error Threshold τ_e

The error threshold τ_e controls the accuracy of the Bézier polygon approximating the outline. When the value of τ_e is reduced, the user requires higher accuracy of the Bézier fitting. Since any Bézier cubic contains at most one inflection point, a single cubic only allows a limited amount of variations. Hence, by adding more control points to split the outline into shorter segments, the specified accuracy is achieved.

To better illustrate the effect of varying the threshold τ_e , we computed in percentage the reduction of the number of control points when the threshold is $\tau_e > 0.5$ compared to that when the threshold is 0.5:

$$\rho(\tau_e) = \frac{\#C(\tau_e) - \#C(0.5)}{\#C(0.5)} \times 100\% , \quad \tau_e > 0.5 . \quad (3.20)$$

Here $\#C(\tau_e)$ denotes the number of control points when the threshold is τ_e . Figure 3.13 (a) shows the average values and the standard deviations of Equation 3.20 when we apply the proposed method to the 20 silhouettes in our data set. We observe that when $\tau_e < 1$, the effect of increasing τ_e is the strongest: the number of control points reduces exponentially. On average, the percentage curves show inflection points around $\tau_e = 1$, that is, when the fitted Bézier polygon has a distance to the sub-pixel outline smaller than 1 pixel. After passing $\tau_e = 1$, increasing τ_e has less impact on the variation of the number of control points. For even larger values of τ_e , there is almost no need to insert new control points, and the corresponding control points are closely related to the corners of the outline. This is justified by the regression in Figure 3.13 (b), where each point represents a silhouette in our data set. It shows that there is a positive relation between the number of corners computed by the Harris-Stephens corner detector [255] and the number of control points when $\tau_e = 10.0$, which is relatively large.

With $\tau_e \gg 1$, the silhouette representation is more compact yet less accurate. With small values of $\tau_e < 1$, we have a more accurate representation yet less efficient. Hence,

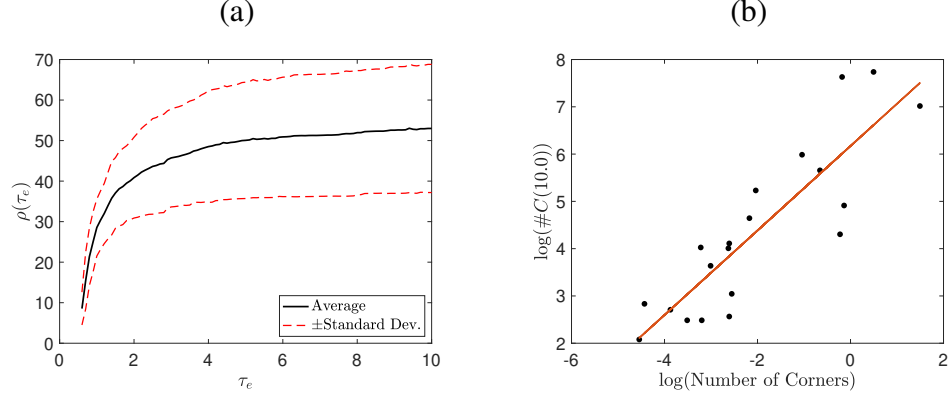


Figure 3.13: (a) For the 20 silhouettes in our data set (Table B.1), the solid curve shows the average relative reduction of the number of control points $\rho(\tau_e)$ Equation 3.20, and the dashed curves indicate the standard deviations. (b) The positive relation between the number of control points when $\tau_e = 10.0$ is large and the number of corners of a silhouette. Each dot represents a sample in our data set. The red curve is computed by linear regression with a goodness of fit $R^2 = 0.75592$.

we would recommend $\tau_e = 1$.

3.9.5 Effect of the Smoothness Parameter σ_0

The smoothness parameter σ_0 adjusts the regularity of the smooth bilinear outline, which approximates ∂S . With larger values of σ_0 , oscillatory features of the given outline are suppressed. With smaller values of σ_0 , the vectorized silhouette preserves sharp corners.

Figure 3.14 demonstrates this effect of σ_0 . We applied the proposed method using $\sigma_0 = 2.0, 1.0$ and 0.5 on the silhouette of a *tree* (a), and the zoom-ins of vectorization results within the boxed region of (a) are presented in (b), (c), and (d), respectively. Observe that the zig-zag around the *tree*'s silhouette is better preserved by reducing σ_0 . As a trade-off, this introduces more control points to recover the sharpness of the outline.

3.9.6 Qualitative Comparison with Feature Point Detectors

Our algorithm produces a set of informative point features of the outline. They include the control points which separate the outline curves into segments for cubic Bézier fitting and the centers of circles. In Figure 3.15, we compare the distribution of these points

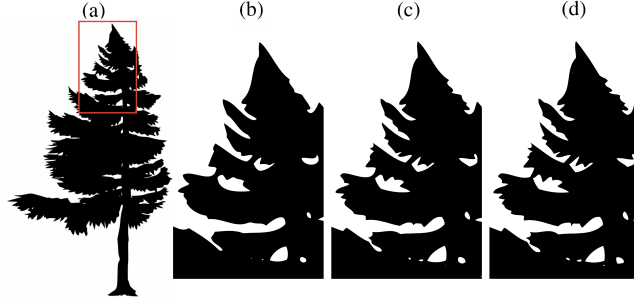


Figure 3.14: Effect of the smoothing parameter σ_0 . (a) A silhouette of a *tree* where the boxed region is examined in detail. Vectorization using (b) $\sigma_0 = 2.0$ (362 control points) (c) $\sigma_0 = 1.0$ (448 control points), and (d) $\sigma_0 = 0.5$ (500 control points). With smaller values of σ_0 , the vectorized outline is sharper, and the number of control points increases.

with the results of some extensively applied feature point detectors: the Harris-Stephens corner detector [255], the features from Accelerated Segment Test (FAST) detector [256], the Speeded Up Robust Features (SURF) detector [257], and the Scale-Invariant Feature Transform (SIFT) [258].

The Harris-Stephens corner detector is a local auto-correlation based method. It locally filters the image with spatial difference operators and identifies corners based on the response. In (a), the Harris-Stephens corner detector identifies all the corners except for the one on the label's right side. The set of control points produced by our algorithm contains all the corners found by the Harris-Stephens detector plus the missed one.

The FAST detector only considers the local configurations of pixel intensities; hence it is widely applied in real-time applications. From (b), we see that FAST identifies all the prominent corners the same as our method. Similarly to (a), there are no FAST points identified around the *balloon*. However, on the circular outline at the center, FAST detects multiple false corners; this illustrates how our algorithm is robust against pixelization.

The SURF detector combines a fast Hessian measure computed via integral images and the distribution of local Haar-wavelet responses to identify feature points that are scale- and translation-invariant. It is similar in that it utilizes the Gaussian scale-space and scale-space interpolation to localize the points of interest. The SURF points are marked over scales; hence we see most of the green crosses in (c) form sequences converging toward the

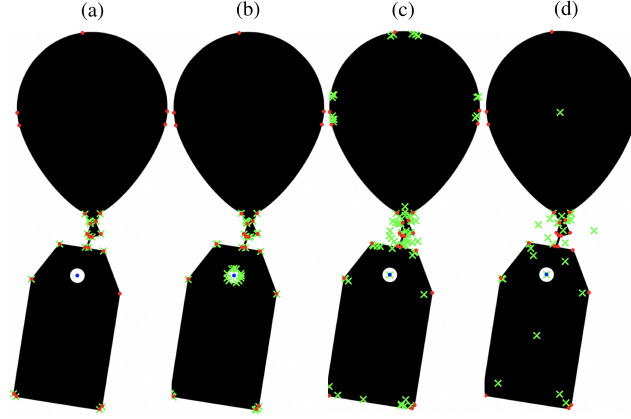


Figure 3.15: Comparison between the control points (red dots) plus the centers of circles (blue dots) produced by the proposed algorithm and other point feature detectors (green crosses). (a) Compared with the Harris corner detector [255]. (b) Compared with the FAST feature detector [256]. (c) Compared with the SURF detector [257]. (d) Compared with the SIFT detector [258]

outline. These limit points correspond precisely to our control points (red dots) distributed over the outline, including those around the *balloon*. Moreover, there is a SURF point at the center of the circular hole in the label, which overlaps with our identified center of the circle (blue dot). Rather than showing feature points over scales, our method locates them directly on the original outline. In (c), notice that our identified points are much simpler compared to SURF points.

SIFT detects scale-invariant features of a given image. As shown in (d), SIFT successfully indicates the presence of corners and marks the *balloon*'s centers as well as the label, which are visually robust features of the silhouette. Our method focuses on the outline instead of the interior points and provides interesting boundary points' locations exactly. Around the *balloon*, the symmetric distribution is compatible with the SIFT point at the center.

The set of control points plus the centers of circles produced by our algorithm is comparable to some of the frequently used feature point detectors in the literature. Hence, in addition to being an effective silhouette vectorization method, the identified control points can be used for other applications where feature point detectors are needed.

3.9.7 Quantitative Comparison with Feature Point Detectors

To further justify that our method can be applied as a stable point feature detector for silhouettes, we compared the techniques discussed above with ours by quantitatively evaluating their performance for the repeatability ratio [259]. This ratio measures the geometric stability of the detected feature points under various transformations.

In particular, for each method, given any angle θ , $0^\circ < \theta < 360^\circ$, we rotated the silhouettes in the first column of Figure 3.16 with respect to their centers by θ respectively, recorded the detected feature points, applied the inverse transform on these points by rotating them back by $-\theta$, then compared their positions with the feature points detected on the original silhouette. Let $n_{\text{repeat}} = 0$. For any rotated feature point, within its ϵ -neighborhood, if we find at least one feature point on the original silhouette, we increase n_{repeat} by 1. The ϵ -repeatability ratio is computed as

$$\frac{n_{\text{repeat}}}{\min\{n_0, n_{\text{transform}}\}} \quad (3.21)$$

where n_0 denotes the number of feature points detected on the original silhouette, and $n_{\text{transform}}$ is the number of feature points detected on the transformed one. During the angle (or scale) changes, this value staying near 1 indicates that the applied method is invariant under rotation (or scale). We fixed $\epsilon = 1.5$ for this experiment.

The second column of Figure 3.16 shows the repeatability ratios under rotations. The set of feature points produced by our method has superior stability when the silhouette is rotated by arbitrary angles. In contrast, the other detectors have low repeatability ratios, especially when the silhouette is turned almost upside-down. Moreover, our method performs consistently well for silhouettes with different geometric features. The *house* silhouette has straight outlines and sharp corners; the *butterfly* silhouette is defined by smooth curves; and the *fish* silhouette has prominent curvature extrema which are not perfect corners.

For the third column of Figure 3.16, we computed the repeatability ratios when the

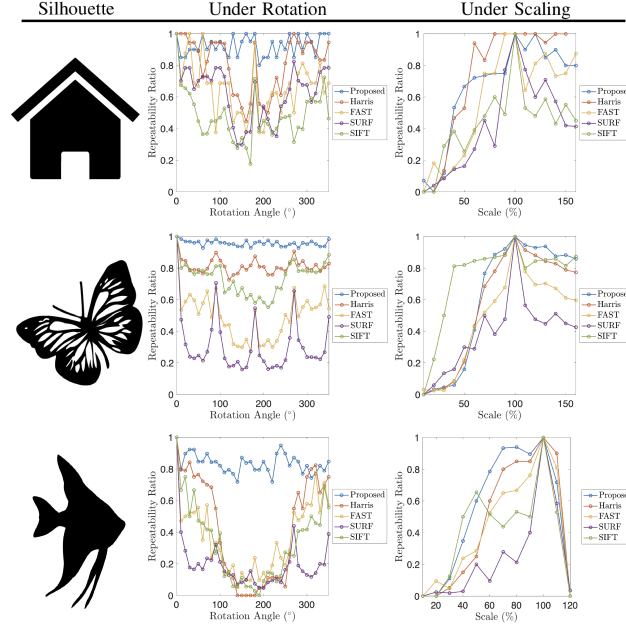


Figure 3.16: Repeatability ratios of the methods in comparison when the silhouettes in the first column are rotated or scaled. Notice that the blue lines (proposed method) are near 1. The performance of our method is the most consistent across these different silhouettes.

transformation is replaced by scaling. Observe that our method is comparable with other detectors, and it is the most consistent one across these different silhouettes.

3.9.8 Comparison with State-of-the-art Software

There are many software available for image vectorization, e.g., Vector Magic [260], Inkspace [261], and Adobe Illustrator 2020 (AI) [262]. In the following set of experiments, we compare our method with these software using the number of control points generated for given silhouettes as a criterion. This quantity is equal to the number of curve segments used for approximating the outline, and a smaller value indicates a more compact silhouette representation.

For comparison, after acquiring SVG files of various silhouettes, we rasterized them and used the PNG images as inputs. Table 3.3 summarizes the results. For Vector Magic, we tested three available settings: high, medium, and low for the vectorization quality. For AI, we chose the setting “Black and White Logo”, as it is suitable for the style of our

inputs. We also include the results when the automatic simplification was used, which are marked by daggers. For Inkspace, we used the default parameter settings. As shown by the mean relative reduction values on the number of control points in the last row, our method produces the most compact vectorization results.

With such an effective reduction in the number of control points, it remains to verify that our method does not over-simplify the representation. We show a detailed comparison in Figure 3.17 between our proposed method and AI. In particular, we used AI without simplification and our method with two sets of parameters: $\sigma_0 = 1$, $\tau_e = 1$ and $\sigma_0 = 0.1$, $\tau_e = 0.5$. We note that σ_0 specifies the smoothness of the recovered outline, and τ_e controls the accuracy. Notice that our method gives fewer control points under these settings, and our results preserve more details of the given silhouettes, for example, the strokes on the scales at the bottom and the sharp outlines on the rear fin.

3.9.9 Quantitative Study of Efficiency and Accuracy

We quantified AI's performance and our method by comparing the given image I and the image I' rasterized from the vectorization result. Denote $S_0 = \{(x, y) \in \Omega \cap \mathbb{N}^2 \mid I(x, y) < 127.5\}$ and $S_r = \{(x, y) \in \Omega \cap \mathbb{N}^2 \mid I'(x, y) < 127.5\}$ as the interior pixels of the given silhouette and the reconstructed one. We evaluated the accuracy of approximating S_0 using S_r by the Dice similarity coefficient [212]

$$\text{DSC} = \frac{2|S_0 \cap S_r|}{|S_0| + |S_r|}. \quad (3.22)$$






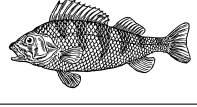
Higher values of DSC ($0 \leq \text{DSC} \leq 1$) imply a better matching between two silhouettes. We evaluated the performance with wide ranges of parameters for both AI and the proposed method. For AI, we tested various combinations of the curve simplification parameter μ (0%–100%) and the corner point angle threshold γ (0°–180°). For our method, we used different combinations of τ_e and σ_0 . Roughly speaking, μ in AI corresponds to τ_e in ours,

which controls the approximating accuracy, and γ in AI corresponds to σ_0 in ours, which adjusts the smoothness of the vectorized outline. Figure 3.18 plots the number of control points against the corresponding DSC values for various parameter settings in both methods. In (a), we fixed the sharpness requirement, i.e., fixed $\gamma = 150^\circ$ (default value for the automatic simplification used in AI) and fixed $\sigma_0 = 1$, and varied μ for AI (the blue curve) and τ_e for ours (the red curve). On the blue curve, larger dots correspond to smaller values of μ ; on the red curve, larger dots correspond to larger values of τ_e . Moving from left to right along both curves indicates more accurate outline approximations. The red curve staying below the blue one, compared to AI, means that our method produces fewer control points while achieving the same level of DSC values. In (b), we present the results of AI using a simplification specified by a set of combinations of parameters ($\gamma = 0^\circ, 10^\circ, \dots, 180^\circ$, $\mu = 0\%, 10\%, \dots, 100\%$). They are organized so that each blue curve corresponds to a fixed value of γ ; higher curves (lighter shades of blue) correspond to larger values of γ while moving from left to right (smaller sizes of dots) along each of the curves corresponds to decreasing μ . The red curve shows our results using different values of τ_e when the merging is applied, and σ_0 is fixed at 0.5. From left to right, the value of τ_e decreases. Observe that the red curve gives a close lower bound for the blue curves when $\text{DSC} < 0.93$. For higher requirements on the accuracy ($\text{DSC} > 0.93$), our method again shows superior efficiency: it requires fewer control points to reach larger DSC values. In contrast, for AI, the best DSC value it can achieve is around 0.95, and adding more control points does not bring any improvement.

3.10 Summary

In this chapter, we considered two types of shape pattern representation via shape skeletons and silhouette vectorization. For the skeleton approach, we discussed the flux-ordered thinning algorithm proposed by [191], which we referred to as the Hamilton-Jacobi Skeleton (HJS), and described an implementation of this method for extracting skeletons of 2D

Table 3.3: Comparison with image vectorization software in terms of the number of control points. We compared with Vector Magic (VM), Inkspace (IS), and Adobe Illustrator 2020 (AI). For VM, we report the number of control points using three settings: High/Medium/Low. For AI, the values with dagger[†] indicate the numbers of control points produced by the automatic simplification. The input image dimensions are 581×564 , 625×598 , 400×390 , 903×499 , 515×529 , and 1356×716 from top to bottom. We also report the mean relative reduction (MRR) of the number of control points computed for the results above.

Test Image	Number of Control Points ($\#C$)				
	Original	VM	IS	AI	Proposed
	405	248/256/245	330	280 (193 [†])	168
	611	359/343/325	383	340 (293 [†])	222
	682	296/294/263	272	211 (128 [†])	120
	1434	915/828/715	932	698 (462 [†])	379
	4434	2789/2582/2370	3292	2120 (1431 [†])	1407
	6664	5470/5218/4955	6493	4870 (3441 [†])	2810
MRR	—	37..97%/40.55%/45.01%	29.88%	45.79% (61.58% [†])	67.38%

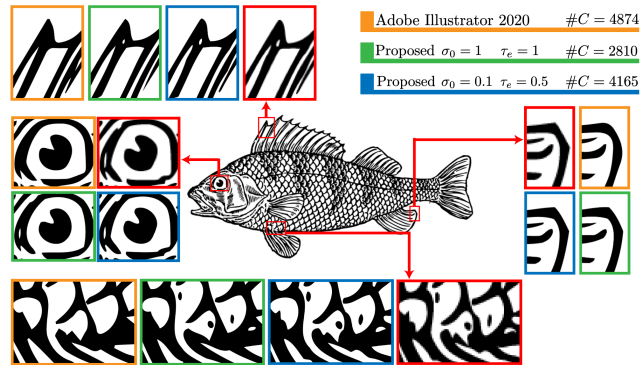


Figure 3.17: Comparison among the given raster image (red boxes), AI (orange boxes), the proposed with $\sigma_0 = 1$, $\tau_e = 1$ (green boxes), and the proposed with $\sigma_0 = 0.1$, $\tau_e = 0.5$ (blue boxes). With smaller numbers of control points ($\#C$), our method preserves better the geometric details of the given silhouette.

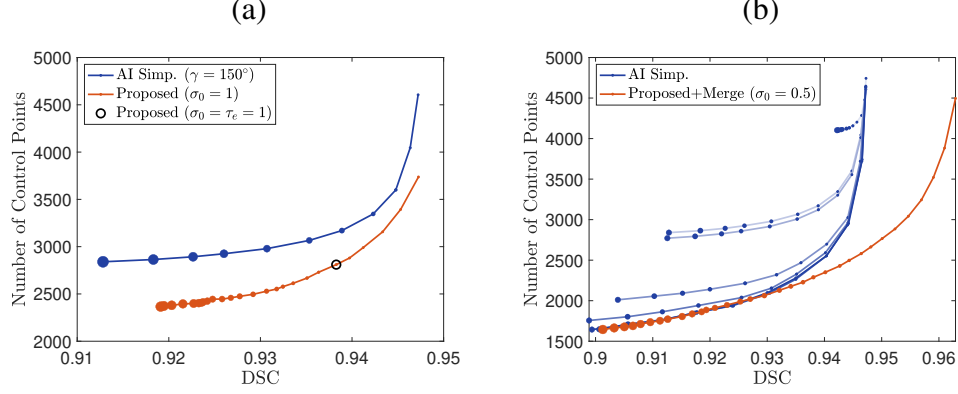


Figure 3.18: (a) Comparison between AI ($\gamma = 150^\circ$) and the proposed method ($\sigma_0 = 1$) when the complexity parameters (μ for AI, τ_e for ours) vary. The circled dot corresponds to our default setting. (b) Comparison between AI with simplification specified by various combinations of μ and γ , and the proposed method using merging with fixed $\sigma_0 = 0.5$ and varying τ_e . In both figures, smaller dots indicate higher levels of complexity for AI (μ) and the proposed method (τ_e), respectively. A dot locating to the right indicates higher accuracy, and a dot in a lower position implies higher efficiency.

shapes from binary images. As a natural extension of the PDE framework, we updated a part of the algorithm computing the distance transformation by the fast sweeping algorithm [202], which improves the efficiency. The robustness of the identified skeleton against boundary perturbations can be adjusted via a single parameter $\gamma > 0$. For arbitrary shapes, we recommend fixing $\gamma = 2.5$ which produces the principal skeleton component and some branches indicating highly irregular features on the boundary. By applying HJS to a fixed shape using varying values of γ , we can obtain a multi-scale shape representation analogous to the approach considered in frequency component analysis. We investigated the special case where $\gamma < 1$ by exploring its connection to the homotopy type of a given shape and illustrating its usage as a deficiency detector for binary shapes. Moreover, we tested the skeleton identified by HJS as a tool for reconstructing shapes from their medial axes. When γ increases, the reconstruction is more precise.

Moreover, we introduced an efficient and effective algorithm for silhouette vectorization [243]. The outline of the silhouette is interpolated bilinearly and uniformly sampled at a sub-pixel level. To reduce the oscillation due to pixelization, we applied the affine short-

ening to the bilinear outline. We identified a set of candidate control points by tracing the curvature extrema across different scales along the well-defined inverse affine shortening flow. This set is then refined by deleting sub-pixel extrema that do not reflect salient corners and inserting new points to guarantee any user-specified accuracy. We also designed special procedures to address the degenerate cases, such as disks, so that our algorithm adapts to arbitrary resolutions and offers better information compression. Our method provides a superior compression ratio by vectorizing the outlines. When the given silhouette undergoes affine transformations, the distribution of control points generated by our method remains relatively stable. These properties are quantitatively justified by the repeatability ratio when compared with popular feature point detectors. Our method is competitive compared to some well-established image vectorization software. It produces results with fewer control points for equally high accuracy. As we saw, the general set up of a vectorization method has been known for a long time and has led to very competitive software. We have adopted this existing set up, but we have verified that applying carefully scale space theory still brings improvements on existing methods. To the best of our knowledge, this yields a first practical evidence that the causality and invariance requirements of scale space theory do lead to better and more accurate shape encoding. This fact has been illustrated by an end-to-end shape rasterization algorithm, which we make public and verifiable on line.

CHAPTER 4

SUBMANIFOLD REPRESENTATION INDUCED BY POINT CLOUD

Acquisition, creation and processing of 3D digital objects is an important topic in various fields, e.g., medical imaging [263], computer graphics [264, 265], industry [266], and preservation of cultural heritage [267]. In industrial and scientific fields [267, 263], surface reconstruction from point cloud data is a critical step in informative data visualization and successful high-level data processing. Effective methods to reconstruct a continuous surface from finitely many points can reduce the burden in data transmission and facilitate shape manipulations. One of the main goals of surface reconstruction is to render a meaningful and reliable surface which captures the geometrical features of the point cloud.

A fundamental step is to reconstruct a surface from a set of point cloud data [268], denoted by $\mathcal{D} \subseteq \mathbb{R}^m$ for $m = 2$ or 3 , such as in Figure 4.1. By the celebrated level set function [269] to represent the surface, for $d \in \mathbb{N}$, a d -dimensional implicit surface is represented by the set

$$\Gamma = \{x \in \mathbb{R}^{d+1} \mid \phi(x) = 0\},$$

for a level set function $\phi : \mathbb{R}^{d+1} \rightarrow \mathbb{R}$. Implicit surfaces enjoy the flexibility in topological changes, and via ϕ , one can easily derive and express geometric features of Γ , such as normals, mean curvature, and Gaussian curvature.

There are a number of related works using implicit surface reconstruction: a data-driven logarithmic prior for noisy data was considered in [270], surface tension was used to enrich the Euler-Lagrange equations in [271], and principal component analysis was used to reconstruct curves which is embedded in sub-manifolds in [272]. In [273], convexified image segmentation model with a fast algorithm was proposed for implicit surface reconstruction for point clouds. In [274], an efficient algorithm for level set method which preserves dis-

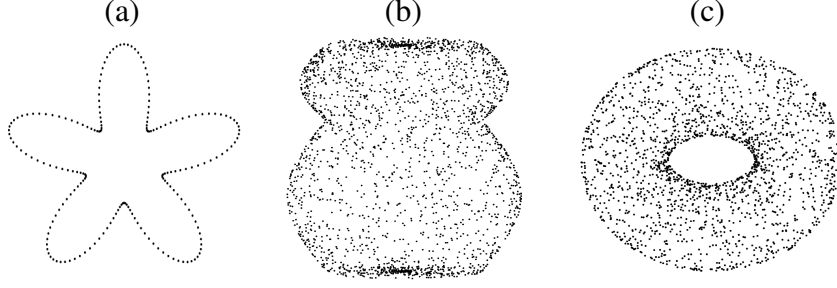


Figure 4.1: Test point clouds. (a) Five-fold circle (200 points). (b) Jar (2100 points). (c) Torus (2000 points).

tance function was proposed. Open surface reconstruction using graph-cuts was proposed in [275], where reconstruction of open surface based on domain decomposition was also proposed. In [276], the authors proposed a variational model consisting of the distance, the normal direction, and the smoothness terms. In [277], a ridge and corner preserving model based on vectorial TV regularization for surface restoration was introduced. In [273], the authors defined the surface via a collection of anisotropic Gaussians centered at each entry of the input point cloud, and used TVG-L1 model for minimization. A similar strategy addressing an ℓ_0 gradient regularization model can be found in [278].

In this chapter, we focus on reconstructing Γ , i.e., a curve in \mathbb{R}^2 or a surface in \mathbb{R}^3 , to represent the underlying structure of the point cloud \mathcal{D} . We will assume only the point locations are given, and no other geometrical information such as normal vectors at each point is known. There are various related works on surface reconstruction from point cloud data: a convection model proposed in [279], a data-driven logarithmic prior for noisy data in [270], using surface tension to enrich the Euler-Lagrange equations in [271], and using principal component analysis to reconstruct curves embedded in sub-manifolds in [272]. A semi-implicit scheme is introduced in [280] to simulate the curvature and surface diffusion motion of the interface. In [273], the authors defined the surface via a collection of anisotropic Gaussians centered at each entry of the input point cloud, and used TVG-L1 model [281] for minimization. A similar strategy addresses an ℓ_0 gradient regularization model proposed in [278]. Some models incorporate additional information. In [276], the

authors proposed a novel variational model, consisting of the distance, the normal, and the smoothness term. Euler’s Elastica model is incorporated for surface reconstruction in [282] where graph cuts algorithm is used. The model in [274] extends the active contours segmentation model to 3D and implicitly allows controlling the curvature of the level set function.

In particular, we describe two variational models, one is solely based on Euclidean distance from the point cloud to the candidate submanifold, and the other considers an additional curvature regularization. Since the resulted functionals are non-convex, we address the optimization challenges by including discussions of the fast algorithms based on operator-splitting and semi-implicit schemes.

4.1 Surface Identification via Minimizing Distance-weighted Surface Area

4.1.1 Energy by Distance-weighted Surface Area

In [279, 203], the authors proposed the minimal surface model by interpreting the reconstructed surface as an elastic membrane attached to the given point cloud. It finds the zero-level set surface Γ that minimizes the following energy:

$$E_s(\Gamma) = \left(\int_{\Gamma} d^s(\mathbf{x}) d\sigma \right)^{\frac{1}{s}} \quad (4.1)$$

Here, $d\sigma$ is the area element, $s > 0$ is an exponent coefficient, $d(\mathbf{x}) = \inf_{\mathbf{y} \in \mathcal{D}} \{|\mathbf{x} - \mathbf{y}|\}$ measures the point-to-point-cloud distance, and \mathcal{D} is the set of point cloud data. The energy (Equation 4.1) minimizes the surface area weighted by the distance from surface to point cloud.

In this section, we will explore fast algorithms to minimize the weighted minimum surface energy (Equation 4.1) for $p = 1$ and 2 proposed in [283]. We describe a Semi-Implicit Method (SIM) to relax the time-step constraint for $p = 2$, and an Augmented Lagrangian Method (ALM) based on the alternating direction method of multipliers (ADMM) ap-

proach for $p = 1$. These algorithms minimize the weighted minimal surface energy (Equation 4.1) with high accuracy and superior efficiency. We analyze the behaviors of ALM in terms of the parameter choices and explore its connection to SIM. Various numerical experiments are presented to discuss the effects of the algorithms.

Let $\Omega \subset \mathbb{R}^m$ ($m = 2$ or 3) denote a bounded domain containing the given point cloud data, \mathcal{D} , a finite set of points. Using the level-set formulation for a codimension 1 submanifold Γ , the d -weighted minimum surface energy (Equation 4.1) can be rewritten as:

$$E_p(\phi) = \left(\int_{\Omega} |d(\mathbf{x})|^p \delta(\phi) |\nabla \phi| \, d\mathbf{x} \right)^{\frac{1}{p}}. \quad (4.2)$$

Here $\delta(x)$ is the Dirac delta function which takes $+\infty$ when $x = 0$, and 0 elsewhere. Compared to Equation 4.1, this integral is defined on Ω , which makes the computation flexible and free from explicitly tracking Γ . We use $p = 2$ for SIM introduced in subsection 4.1.2, and $p = 1$ for ALM in subsection 4.1.3. In general, $p = 2$ is a natural choice, since it provides better stability and efficiency for a semi-implicit type PDE-based method. For ALM, we explore $p = 1$ to take advantage of an aspect of fast algorithm in ADMM setting such as shrinkage, similarly to the case in [284]. Visually, the numerical results of surface reconstruction are similar for $p = 1$ or $p = 2$ (see section 4.2).

4.1.2 Semi-Implicit Method (SIM)

We introduce a gradient-flow-based semi-implicit method to minimize

$$E_2(\phi) = \left(\int_{\Omega} d(\mathbf{x})^2 \delta(\phi) |\nabla \phi| \, d\mathbf{x} \right)^{\frac{1}{2}}. \quad (4.3)$$

Following [285], the first variation of $E_2(\phi)$ with respect to ϕ is characterized as a functional:

$$\begin{aligned} \left\langle \frac{\partial E_2(\phi)}{\partial \phi}, v \right\rangle &= - \int_{\Omega} \frac{1}{2} \delta(\phi) \left[\int_{\Omega} d^2(\mathbf{x}) \delta(\phi) |\nabla \phi| d\mathbf{x} \right]^{-1/2} \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] v d\mathbf{x} \\ &+ \int_{\partial\Omega} \frac{d(\mathbf{x})^2 \delta(\phi)}{|\nabla \phi|} (\nabla \phi \cdot \mathbf{n}) v d\mathbf{x} \end{aligned}$$

for any test function v from the Sobolev space H^1 where \mathbf{n} denotes the outward normal direction along $\partial\Omega$. Minimizing Equation 4.3 is equivalent to finding the critical point ϕ such that $\left\langle \frac{\partial E_2(\phi)}{\partial \phi}, v \right\rangle = 0, \forall v \in H^1$. This is associated with solving the following initial value problem:

$$\begin{cases} \frac{\partial \phi}{\partial t} = \bar{f}(d, \phi) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] \text{ in } \Omega, \\ \frac{d(\mathbf{x})^2 \delta(\phi)}{|\nabla \phi|} \frac{\partial \phi}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \\ \phi(\mathbf{x}, 0) = \phi^0, \end{cases} \quad (4.4)$$

where ϕ^0 is an initial guess for the unknown ϕ , and $\bar{f}(d, \phi) = \frac{1}{2} \delta(\phi) \left[\int_{\Omega} d^2(\mathbf{x}) \delta(\phi) |\nabla \phi| d\mathbf{x} \right]^{-1/2}$. The steady state solution of Equation 4.4 gives a minimizer ϕ^* of $E_2(\phi)$.

Remark 4.1.1. *Since we focus on the zero level set of ϕ , to make our scheme more stable, we apply a reinitialization to ϕ after every several iterations which modifies ϕ to be a signed distance function while keeping the location of the zero level set. See subsection 4.1.5 for more details. Consequently, the effect of the boundary condition is negligible. For the computational efficiency (e.g., applying the Fast Fourier Transform), we replace the boundary condition of ϕ in Equation 4.4 by a periodic boundary condition.*

Here the delta function δ is realized as the derivative of the one dimensional Heaviside function $H : \mathbb{R} \rightarrow \{0, 1\}$. We adopt the smooth approximation of $H(\phi)$ as in [286]:

$$H(\phi) \approx H_{\varepsilon}(\phi) = \frac{1}{2} + \arctan(\phi/\varepsilon)/\pi \quad \text{and} \quad \delta(\phi) \approx H'_{\varepsilon}(\phi) = \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} \quad (4.5)$$

with $\varepsilon > 0$ as the smoothness parameter. Then \bar{f} is approximated by its smoothed version f expressed as

$$f(d, \phi) = \frac{1}{2} \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} \left[\int_{\Omega} d^2(\mathbf{x}) \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} |\nabla \phi| d\mathbf{x} \right]^{-1/2}.$$

We add a stabilizing diffusive term $-\beta \Delta \phi$ for $\beta > 0$ on both sides of the PDE in Equation 4.4 to consolidate the computation, similarly to [280]:

$$\frac{\partial \phi}{\partial t} - \beta \Delta \phi = -\beta \Delta \phi + f(d, \phi) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right]. \quad (4.6)$$

Employing a semi-implicit scheme, we solve ϕ from Equation 4.6 by iteratively updating ϕ^{n+1} using ϕ^n via the following equation:

$$\frac{\phi^{n+1}}{\Delta t} - \beta \Delta \phi^{n+1} = \frac{\phi^n}{\Delta t} - \beta \Delta \phi^n + f(d, \phi^n) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi^n}{|\nabla \phi^n|} \right], \quad (4.7)$$

where Δt is the time-step. This equation can be efficiently solved by the Fast Fourier Transform (FFT). Denoting the discrete Fourier transform by \mathcal{F} and its inverse by \mathcal{F}^{-1} , we have

$$\mathcal{F}(\phi)(i \pm 1, j) = e^{\pm 2\pi\sqrt{-1}(i-1)/M} \mathcal{F}(\phi)(i, j), \quad \mathcal{F}(\phi)(i, j \pm 1) = e^{\pm 2\pi\sqrt{-1}(j-1)/N} \mathcal{F}(\phi)(i, j).$$

Accordingly, the discrete Fourier transform of $\Delta \phi$ is

$$\mathcal{F}(\Delta \phi)(i, j) = [2 \cos(\pi\sqrt{-1}(i-1)/M) + 2 \cos(\pi\sqrt{-1}(j-1)/N) - 4] \mathcal{F}(\phi)(i, j).$$

Here the coefficient in front of $\mathcal{F}(\phi)(i, j)$ represents the diagonalized discrete Laplacian operator in the frequency domain. Let g_1 be the right side of Equation 4.7, then the solution

$\phi^{n+1}(i, j)$ of Equation 4.7 is computed via

$$\phi^{n+1}(i, j) = \mathcal{F}^{-1} \left(\frac{\mathcal{F}(g_1)(i, j)}{(1 - \beta \Delta t [2 \cos(\pi \sqrt{-1}(i-1)/N) + 2 \cos(\pi \sqrt{-1}(j-1)/N) - 4])} \right). \quad (4.8)$$

As for the stopping criterion, we exploit the mean relative change of the weighted minimum surface energy (Equation 4.1). At the n^{th} iteration, the algorithm terminates if

$$\frac{|\bar{e}_{n-1}^k - \bar{e}_n^k|}{\bar{e}_n^k} < 10^{-4}, \quad \text{where} \quad \bar{e}_n^k = \frac{1}{k} \sum_{i=n-k}^n E_p(\phi^i). \quad (4.9)$$

Here the quantity \bar{e}_n^k represents the average of the energy values computed from the $(n-k)^{\text{th}}$ to the n^{th} iteration for some $k \in \mathbb{N}$, $k \geq 1$. We fix $k = 10$ and set $p = 2$ for SIM. We summarize the main steps of SIM in Algorithm algorithm 5.

Initialization: d, ϕ^0 and $n = 0$.

while *the stopping criterion* (Equation 4.9) *with* $p = 2$ *is greater than* 10^{-4} **do**

Update ϕ^{n+1} from ϕ^n solving Equation 4.8;

Update $n \leftarrow n + 1$;

end

Output: ϕ^n such that $\{\phi^n = 0\}$ approximates $\{\phi^* = 0\}$.

Algorithm 5: SIM for the weighted minimum surface (Equation 4.3)

4.1.3 Augmented Lagrangian Method (ALM)

In this section, we present an augmented Lagrangian-based method to minimize the weighted minimum surface energy (Equation 4.2) for $p = 1$, i.e.,

$$E_1(\phi) = \int_{\Omega} d(\mathbf{x}) \delta(\phi) |\nabla \phi| \, d\mathbf{x}. \quad (4.10)$$

For the non-differentiable term $|\nabla \phi|$ in (Equation 4.10), we utilize the variable-splitting technique and introduce an auxiliary variable $\mathbf{p} = \nabla \phi$. We rephrase the minimization of

$E_1(\phi)$ as a constrained optimization problem:

$$\{\phi^*, \mathbf{p}^*\} = \arg \min_{\phi, \mathbf{p}} \int_{\Omega} \frac{\varepsilon d|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} d\mathbf{x}, \quad \text{subject to } \mathbf{p} = \nabla \phi, \quad (4.11)$$

here we replace $\delta(\phi)$ by its smooth approximation $H'_\varepsilon(\phi)$ as in (Equation 4.5). To solve problem (Equation 4.11), we formulate the augmented Lagrangian function:

$$\mathcal{L}(\phi, \mathbf{p}, \boldsymbol{\lambda}; r) = \int_{\Omega} \frac{\varepsilon d|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} d\mathbf{x} + \frac{r}{2} \int_{\Omega} |\mathbf{p} - \nabla \phi|^2 d\mathbf{x} + \int_{\Omega} \boldsymbol{\lambda} \cdot (\mathbf{p} - \nabla \phi) d\mathbf{x}, \quad (4.12)$$

where $r > 0$ is a scalar penalty parameter and $\boldsymbol{\lambda} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ represents the Lagrangian multiplier. Minimizing Equation 4.12 amounts to considering the following saddle-point problem:

$$\begin{aligned} & \text{Find } (\phi^*, \mathbf{p}^*, \boldsymbol{\lambda}^*) \in \mathbb{R} \times \mathbb{R}^m \times \mathbb{R}^m \\ & \text{s.t. } \mathcal{L}(\phi^*, \mathbf{p}^*, \boldsymbol{\lambda}; r) \leq \mathcal{L}(\phi^*, \mathbf{p}^*, \boldsymbol{\lambda}^*; r) \leq \mathcal{L}(\phi, \mathbf{p}, \boldsymbol{\lambda}^*; r); \\ & \forall (\phi, \mathbf{p}, \boldsymbol{\lambda}) \in \mathbb{R} \times \mathbb{R}^m \times \mathbb{R}^m. \end{aligned} \quad (4.13)$$

Given ϕ^n , \mathbf{p}^n , and $\boldsymbol{\lambda}^n$, for $n = 0, 1, 2, \dots$, the $(n + 1)^{\text{th}}$ iteration of an ADMM-type algorithm for Equation 4.13 consists of solving a series of sub-problems:

$$\phi^{n+1} = \arg \min_{\phi} \mathcal{L}(\phi, \mathbf{p}^n, \boldsymbol{\lambda}^n; r); \quad (4.14)$$

$$\mathbf{p}^{n+1} = \arg \min_{\mathbf{p}} \mathcal{L}(\phi^{n+1}, \mathbf{p}, \boldsymbol{\lambda}^n; r); \quad (4.15)$$

$$\boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^n + r (\mathbf{p}^{n+1} - \nabla \phi^{n+1}). \quad (4.16)$$

Each sub-problem can be solved efficiently. First, we find the minimizer of the sub-problem (Equation 4.14) by solving its Euler-Lagrange equation:

$$-r \Delta \phi^{n+1} = \frac{2d\varepsilon |\mathbf{p}^n| \phi^n}{\pi(\varepsilon^2 + (\phi^n)^2)^2} - \nabla \cdot (r \mathbf{p}^n + \boldsymbol{\lambda}^n). \quad (4.17)$$

Here Δ is the Laplacian operator. Following [284], we introduce a frozen-coefficient term $\eta\phi$, for $\eta > 0$, on both sides of Equation 4.17 to stabilize the computation; thus, Equation 4.14 is solved using the following equation:

$$\eta\phi^{n+1} - r\Delta\phi^{n+1} = \eta\phi^n + \frac{2d\varepsilon|\mathbf{p}^n|\phi^n}{\pi(\varepsilon^2 + (\phi^n)^2)^2} - \nabla \cdot (r\mathbf{p}^n + \boldsymbol{\lambda}^n). \quad (4.18)$$

We solve this via FFT, similarly to Equation 4.8 for SIM. Thus, the ϕ sub-problem is solved via

$$\phi^{n+1}(i, j) = \mathcal{F}^{-1} \left(\frac{\mathcal{F}(g_2)(i, j)}{(\eta - r [2 \cos(\pi\sqrt{-1}(i-1)/N) + 2 \cos(\pi\sqrt{-1}(j-1)/N) - 4])} \right). \quad (4.19)$$

Second, the \mathbf{p} sub-problem (Equation 4.15) is equivalent to a weighted Total Variation (TV) minimization, whose solution admits a closed-form expression using the shrinkage operator [287]. Explicitly, the updated \mathbf{p}^{n+1} is computed via:

$$\mathbf{p}^{n+1} = \max \left\{ 0, 1 - \frac{d\varepsilon}{\pi(\varepsilon^2 + (\phi^{n+1})^2)|r\nabla\phi^{n+1} - \boldsymbol{\lambda}^n|} \right\} \left(\nabla\phi^{n+1} - \frac{\boldsymbol{\lambda}^n}{r} \right). \quad (4.20)$$

Finally, the Lagrangian multiplier $\boldsymbol{\lambda}$ is updated by Equation 4.16. The stopping criterion for the ALM iteration is the same as that for SIM (Equation 4.9), but with $p = 1$. We summarize the main steps of ALM in algorithm 6.

Input: Set $d, \phi^0, \mathbf{p}^0, \boldsymbol{\lambda}^0$, and $n = 0$.
while the stopping criterion (Equation 4.9) with $p = 1$ is greater than 10^{-4} **do**
 Update $\phi^{n+1} = \arg \min_{\phi} \mathcal{L}(\phi, \mathbf{p}^n, \boldsymbol{\lambda}^n; r)$ via (Equation 4.19) ;
 Update $\mathbf{p}^{n+1} = \arg \min_{\mathbf{p}} \mathcal{L}(\phi^{n+1}, \mathbf{p}, \boldsymbol{\lambda}^n; r)$ via (Equation 4.20);
 Update $\boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^n + r(\mathbf{p}^{n+1} - \nabla\phi^{n+1})$;
 Update $n \leftarrow n + 1$;
end
Output: ϕ^n such that $\{\phi^n = 0\}$ approximates $\{\phi^* = 0\}$.

Algorithm 6: ALM for the weighted minimum surface (Equation 4.10)

4.1.4 Connection between SIM and ALM Algorithms

Note that both SIM and ALM involve solving elliptic PDEs of the form:

$$a\phi - b\Delta\phi = g, \quad (4.21)$$

for some constants $a, b > 0$, and a function g defined on Ω . For SIM, it is Equation 4.7:

$$\underbrace{\frac{1}{\Delta t}}_a \phi^{n+1} - \underbrace{\beta}_b \Delta\phi^{n+1} = \underbrace{\frac{\phi^n}{\Delta t} - \beta\Delta\phi^n + f(d, \phi^n)\nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla\phi^n}{|\nabla\phi^n|} \right]}_g,$$

and for ALM, it is Equation 4.18:

$$\underbrace{\eta}_a \phi^{n+1} - \underbrace{r}_b \Delta\phi^{n+1} = \underbrace{\eta\phi^n + \frac{2d\varepsilon|\mathbf{p}^n|\phi^n}{\pi(\varepsilon^2 + (\phi^n)^2)} - \nabla \cdot (r\mathbf{p}^n + \boldsymbol{\lambda}^n)}_g.$$

We remark interesting connections between SIM and ALM. First, both methods have stabilizing terms but in different positions on the left side of Equation 4.21. For SIM, it is $-\beta\Delta\phi$, while for ALM, it is $\eta\phi$. Second, relating the coefficients of ϕ , $1/\Delta t$ in SIM gives insight to the effect of η in ALM. In general, a large η slows down the convergence of ALM, while a small η accelerates it (as the effect of $1/\Delta t$ on SIM). Figure 4.2 shows convergence behaviors of ALM for different η , using the five-fold circle point cloud in Figure 4.1 (a). It displays the CPU time (in seconds) for $r = 1$, $\varepsilon = 1$, and η varying from 0.05 to 0.5. Note that as η increases, the time required to reach the convergence increases almost quadratically at first, then stays around the same level. Third, the correspondence between $b = \beta$ in SIM, and $b = r$ in ALM allows another interpretation of the parameter r in ALM. In SIM, a large β smears the solution and avoids discontinuities or sharp corners, and for ALM, large r also allows to pass through fine details. Figure 4.7 in Section section 4.2 presents more details, where we experiment with different r and ε values for the five-fold

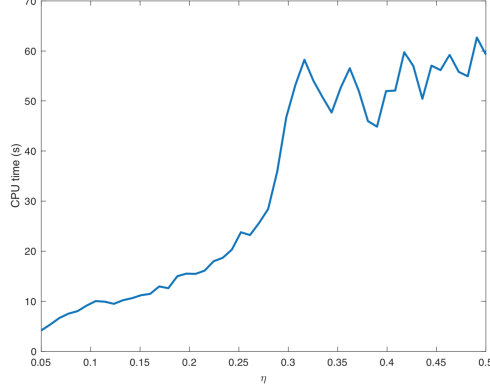


Figure 4.2: The CPU-time (s) of ALM until convergence for the five-fold circle point cloud in Figure 4.1 (a). Here $r = \varepsilon = 1$ and η varies from 0.05 to 0.5. The connection between SIM and ALM indicates that large η slows down ALM. In this graph, as η increases, the time required to reach the convergence increases.

circle point cloud shown in Figure 4.1 (a).

4.1.5 Implementation Details

We illustrate the details for planar point clouds, i.e., $\mathcal{D} \subseteq \mathbb{R}^2$, and the extension to \mathbb{R}^3 is straightforward. Let the computational domain $\Omega = [0, M] \times [0, N]$, $M, N > 0$, be discretized by a Cartesian grid with $\Delta x = \Delta y = 1$. For any function u (or a vector field $\mathbf{v} = (v^1, v^2)$) defined on Ω , we use $u_{i,j}$ or $u(i, j)$ to denote $u(i\Delta x, i\Delta y)$. We use the usual backward and forward finite difference schemes:

$$\begin{aligned} \partial_1^- u_{i,j} &= \begin{cases} u_{i,j} - u_{i-1,j}, & 1 < i \leq M; \\ u_{1,j} - u_{M,j}, & i = 1. \end{cases} & \partial_1^+ u_{i,j} &= \begin{cases} u_{i+1,j} - u_{i,j}, & 1 \leq i < M - 1; \\ u_{1,j} - u_{M,j}, & i = M. \end{cases} \\ \partial_2^- u_{i,j} &= \begin{cases} u_{i,j} - u_{i,j-1}, & 1 < j \leq N; \\ u_{i,1} - u_{i,N}, & j = 1. \end{cases} & \partial_2^+ u_{i,j} &= \begin{cases} u_{i,j+1} - u_{i,j}, & 1 \leq j < N - 1; \\ u_{i,1} - u_{i,N}, & j = N. \end{cases} \end{aligned}$$

The gradient, divergence and the Laplacian operators are approximated as follows:

$$\begin{aligned}\nabla u_{i,j} &= ((\partial_1^- u_{i,j} + \partial_1^+ u_{i,j})/2, (\partial_2^- u_{i,j} + \partial_2^+ u_{i,j})/2); \\ \nabla \cdot \mathbf{v}_{i,j} &= (\partial_1^+ v_{i,j}^1 + \partial_1^- v_{i,j}^1)/2 + (\partial_2^+ v_{i,j}^2 + \partial_2^- v_{i,j}^2)/2; \\ \Delta u_{i,j} &= \partial_1^+ u_{i,j} - \partial_1^- u_{i,j} + \partial_2^+ u_{i,j} - \partial_2^- u_{i,j}.\end{aligned}$$

The distance function d is computed once at the beginning and no update is needed. It satisfies an Eikonal equation:

$$\begin{cases} |\nabla d| = 1 \text{ in } \Omega, \\ d(\mathbf{x}) = 0 \text{ for } \mathbf{x} \in \mathcal{D}, \end{cases} \quad (4.22)$$

and discretizing Equation 4.22 via the Lax-Friedrich scheme leads to an updating formula:

$$d_{i,j}^{n+1} = \frac{1}{2} \left(1 - |\nabla d_{i,j}^n| + \frac{d_{i+1,j}^n + d_{i-1,j}^n}{2} + \frac{d_{i,j+1}^n + d_{i,j-1}^n}{2} \right). \quad (4.23)$$

We solve Equation 4.23 using the fast sweeping method [288] with complexity $O(G)$ for G grid points.

Keeping ϕ^n to be a signed distance function during the iteration improves the stability of level-set-based algorithms. We reinitialize ϕ^n at the n^{th} iteration by solving the following PDE:

$$\begin{cases} \phi_\tau + \text{sign}(\phi)(|\nabla \phi| - 1) = 0, \\ \phi(\mathbf{x}, 0) = \phi^n. \end{cases} \quad (4.24)$$

Here the subscript τ represents the partial derivative with respect to an artificial time, and $\text{sign} : \mathbb{R} \rightarrow \{-1, 0, 1\}$ is the sign function. We discretize Equation 4.24 via an explicit

time Lax-Friedrichs scheme. For $k = 0, 1, \dots, K$, we update

$$\phi_{i,j}^{(k+1)} = \phi_{i,j}^{(k)} - \Delta\tau \operatorname{sign} \left[(\phi_{i,j}^{(k)}) (|\nabla \phi_{i,j}^{(k)}| - 1) - \frac{\phi_{i-1,j}^k + \phi_{i+1,j}^k + \phi_{i,j-1}^k + \phi_{i,j+1}^k - 4\phi_{i,j}^k}{2} \right], \quad (4.25)$$

with $\phi_{i,j}^{(0)} = \phi_{i,j}^n$. In practice, ϕ^n being a signed distance function near the 0-level-set is important; thus, it is sufficient to evolve Equation 4.25 for a small K and update ϕ^n with $\phi^{(K)}$. We fix $K = 10$ throughout this paper.

For many problems, various spacial resolutions, i.e., different values for Δx and Δy , may be needed. We allow this flexibility by scaling the data up (or down) to some level such that $\Delta x = \Delta y = 1$ is sufficient; then we transform the reconstructed surface back to the original scale. Hence, the accuracy depends on the density of the rescaled point cloud data.

4.2 Numerical Experiments on Model of Distance-weighted Surface Area

4.2.1 General Performance on 2D and 3D Point Clouds

For both SIM and ALM, we vary the value of ε from 0.5 to 1. For SIM, we use $\Delta t = 500$. When the point cloud \mathcal{D} is in 2D, we set $\beta = 0.1$, and when D is in 3D, $\beta = 0.01$. For ALM, the value of η ranges from 0.05 to 1, and r from 0.5 to 2.

The code is written in MATLAB and executed without additional machine support, e.g. parallelization or GPU-enhanced computations. All the experiments are performed on Intel® Core™4-Core 1.8GHz (4.0GHz with Turbo) machine, with 16 GB/RAM and Intel® UHD Graphics 620 graphic card under Windows OS. The contours and isosurfaces are displayed using MATLAB visualization engine. No post-processing, e.g., smoothing or sharpening, is applied.

For our first experiment, Figure 4.3 displays a set of planar curves reconstructed from 2D point clouds confined within a square $\Omega = [0, 100]^2 \subset \mathbb{R}^2$. We generate the data using four different shapes: a triangle, an ellipse, a square whose corners are missing, and a five-

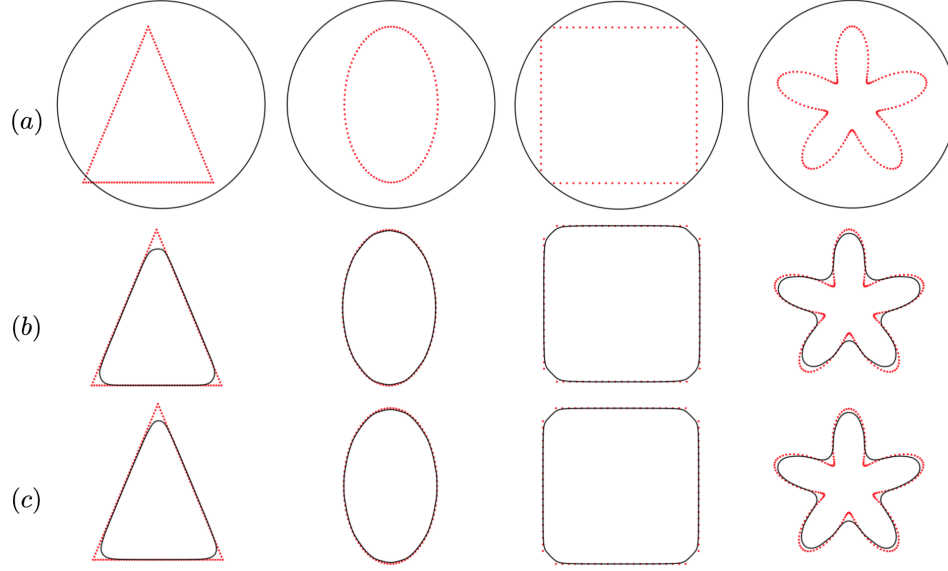


Figure 4.3: The test point clouds: triangle with 150 number of points, ellipse with 100 points, square with 80 points, and five-fold-circle with 200 points. (a) The top row, identical initial condition applied to SIM and ALM for different \mathcal{D} . (b) The middle row, the results obtained by SIM. (c) The bottom row, the results obtained by ALM using $r = 1.5$. Both methods give compatible results.

fold circle. For these cases, we use a centered circle with radius 30 as the initial guess, shown in Figure 4.3 (a). Figure 4.3 (b) and (c) display the given \mathcal{D} , as well as the curves identified by SIM and ALM with $r = 1.5$, respectively. Both methods produce comparably accurate results. In the triangle example, corners get as close as the approximated delta function (with parameter ε) allows for both methods. The ellipse and square results fit very closely to the respective point clouds. For the five-fold-circle, there is a slight difference in how the curve fits the edges, yet the results are very compatible.

Table 4.1 shows the CPU times (in seconds) for SIM, ALM using $r = 0.5, 1, 1.5$, and 2, as well as the times for the explicit method in [285] using $\Delta t = 20$ on the same data sets. With proper choices of r , ALM outperforms the other methods in terms of computational efficiency. SIM is stable without any dependency on the choice of parameters, and its run-times are comparable to the best performances of ALM in most cases. Both methods are faster than the explicit method in all the examples.

The second set of experiments reconstruct surfaces from the point clouds in 3D: a jar

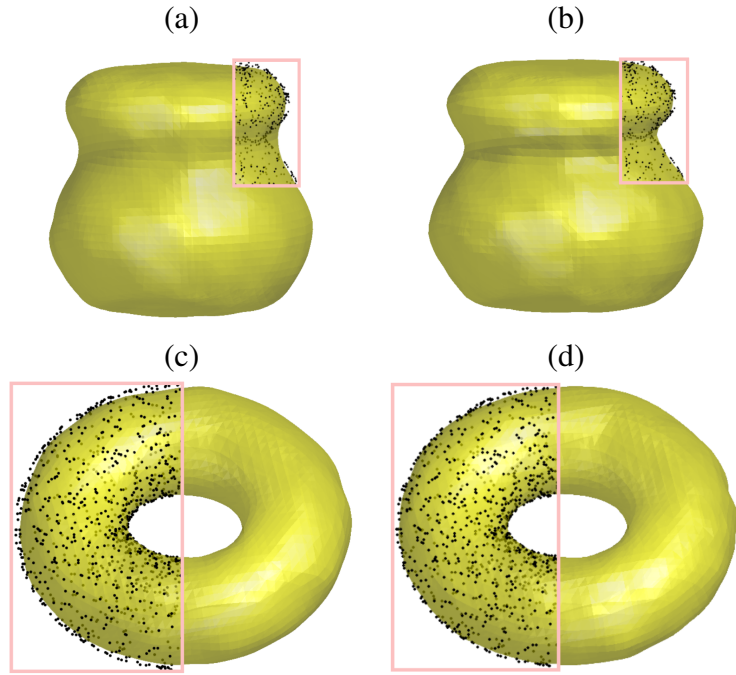


Figure 4.4: The first row shows ALM and SIM applied to the 3D jar point cloud in Figure 4.1 (b). (a) The result of ALM with $r = 1.3, \varepsilon = 0.5, \eta = 0.6$. (b) The result of SIM. The second row shows the methods applied to the 3D torus point cloud in Figure 4.1 (c). (c) The result of ALM with $r = 1.3, \varepsilon = 0.5, \eta = 0.6$. (d) The result of SIM. Both methods are compatible and shows good results.

Table 4.1: CPU time (s) for SIM, ALM using $r = 0.5, 1, 1.5$, and 2, and the explicit method in [285] with $\Delta t = 20$ for the point cloud data sets in Figure 4.3. Both SIM and ALM shows fast convergence.

Object	ALM($r = 0.5$)	ALM($r = 1$)	ALM($r = 1.5$)	ALM($r = 2$)	SIM	[285]
Triangle	—	1.45	1.31	1.48	1.50	5.25
Ellipse	1.22	1.03	1.33	1.37	1.49	3.89
Square	—	—	0.94	1.20	1.09	2.07
Five-fold circle	0.83	1.44	1.86	1.22	1.96	4.18

in Figure 4.1 (b) and a torus in Figure 4.1 (c) within $\Omega = [0, 50]^3$. In Figure 4.4, we show the reconstructed surfaces using SIM and ALM. A portion of the given point cloud is superposed for validation in each case. Both methods successfully capture the overall shapes and non-convex features of the jar, as well as the torus. There are only slight differences in the reconstruction between using SIM with $p = 2$ and using ALM with $p = 1$.

Table 4.2 shows the efficiency of SIM and ALM compared to the explicit method in [285] for the experiments in Figure 4.4. Thanks to the semi-implicit scheme, the time step can be large and we used $\Delta t = 500$ in SIM; in the explicit method, we are forced to use much smaller time step $\Delta t = 20$ to maintain the stability. The improvement of run-time in ALM is carefully controlled by the parameters r, ε and η . We choose $r = 1.3, \varepsilon = 0.5$ and $\eta = 0.6$ for both cases. Both SIM and ALM efficiently provide accurate reconstructions.

Table 4.2: CPU time (s) of SIM and ALM compared to the explicit method in [285] for the point cloud data sets of Figure 4.4. Both SIM and ALM show fast convergence.

Object	ALM	SIM	[285]
Jar	29.69	29.42	74.44
Torus	47.32	33.58	114.20

The third set of examples show the effect of the distance function d . Notice that the weighted minimal surface energy (Equation 4.1) is mainly driven by the distance function d , that is, the given point cloud \mathcal{D} determines the landscape of d , which affects the behavior of the level-set during the evolution. Figure 4.5 shows the evolution using ALM, applied to different subsets of point clouds sampled from the same bunny face shape. The

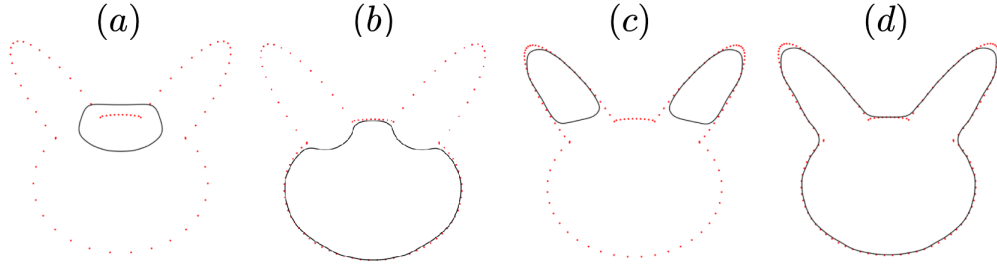


Figure 4.5: The effect of the distance function for varying-density point clouds: the face with n_1 points, the head with n_2 points, and each ear with n_3 points. (a) the given point cloud is with $(n_1, n_2, n_3) = (20, 10, 20)$, and shows the 0-level-set of ϕ^n at 15th iteration, (b) $(n_1, n_2, n_3) = (50, 10, 20)$, and shows 18th iteration, and (c) $(n_1, n_2, n_3) = (20, 10, 40)$, and shows 20th iteration. These three curves eventually degenerate to a point. (d) is with $(n_1, n_2, n_3) = (50, 10, 40)$ and shows the converged solution. The potential energy (Equation 4.1) is mainly driven by the distance function d , which affects the level-set evolution.

densities of the point cloud vary for the three different regions: the face with n_1 points, the head with n_2 points, and each ear with n_3 points. Figure 4.5 (a) shows the given point cloud for $(n_1, n_2, n_3) = (20, 10, 20)$, with the 0-level-set of ϕ^n at 15th iteration, (b) for $(n_1, n_2, n_3) = (50, 10, 20)$, at 18th iteration, and (c) for $(n_1, n_2, n_3) = (20, 10, 40)$, at 20th iteration. These three curves eventually degenerate to a point, since the energy model (Equation 4.2) drives curves to have short lengths, i.e., the level set tends to shrink. (d) for $(n_1, n_2, n_3) = (50, 10, 40)$ and shows the converged solution. In (a)–(c), denser parts of the point cloud attract the curve with stronger forces, and the sparser parts of the point cloud fail to lock the curve. In (d), with a more balanced distribution of points, the curve converges to correct shape.

The fourth set of examples demonstrates the robustness of ALM and SIM against noise. Figure 4.6 shows the reconstructed curves from clean and noisy data: (a)–(c) are results of ALM, and (d)–(f) are results of SIM. (a) and (d) in the first column show results obtained from the clean data, which has 200 points sampled from a three-fold circle. Gaussian noise with standard deviation 1 is added to both x and y coordinates to generate noisy point cloud in the second column, (b) and (e). To show the differences, the third column superposes both results reconstructed from clean and noisy point clouds. Both ALM and

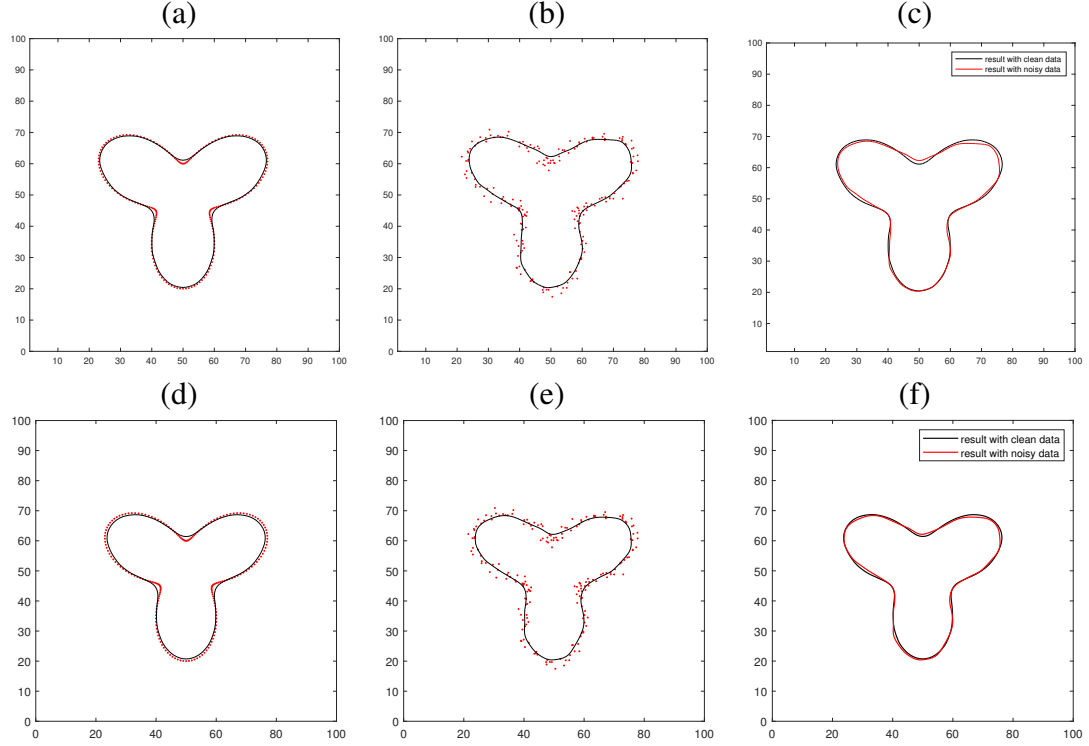


Figure 4.6: The influence of noise on reconstructing three-fold circle with 200 points: (a)-(c) ALM and (d)-(f) SIM. The first column shows the reconstructed curves from clean data, and the second column the reconstructions from noisy data. The third column shows the comparison between the two reconstructed curves in first two columns.

SIM provide compatible results. For the noisy data, although the reconstructed curves show some oscillation, they are very close to the solutions using the clean data, respectively.

4.2.2 Choice of Parameters for ALM and the Effects

The proposed ALM has one parameter $r > 0$, and the model (Equation 4.2) uses the delta function, where the smoothness parameter $\varepsilon > 0$ is added to stabilize the computation. Both parameters have straightforward effects on the level-set evolution from Equation 4.17. For example, consider a set of points within a thin-band around the 0-level-set of ϕ^n , denoted by $B_\varepsilon = \{\mathbf{x} \mid -2\varepsilon/\sqrt{3} < \phi^n(\mathbf{x}) < 2\varepsilon/\sqrt{3}\}$. By the continuity of ϕ^n , there exist \mathbf{y} and $\mathbf{z} \in B_\varepsilon$ such that $\phi^n(\mathbf{y}) = -\varepsilon/\sqrt{3}$ and $\phi^n(\mathbf{z}) = \varepsilon/\sqrt{3}$; these values are the minimum and maximum of the function $h(x) = \frac{2\varepsilon x}{\pi(\varepsilon^2 + x^2)^2}$, respectively. At these points,

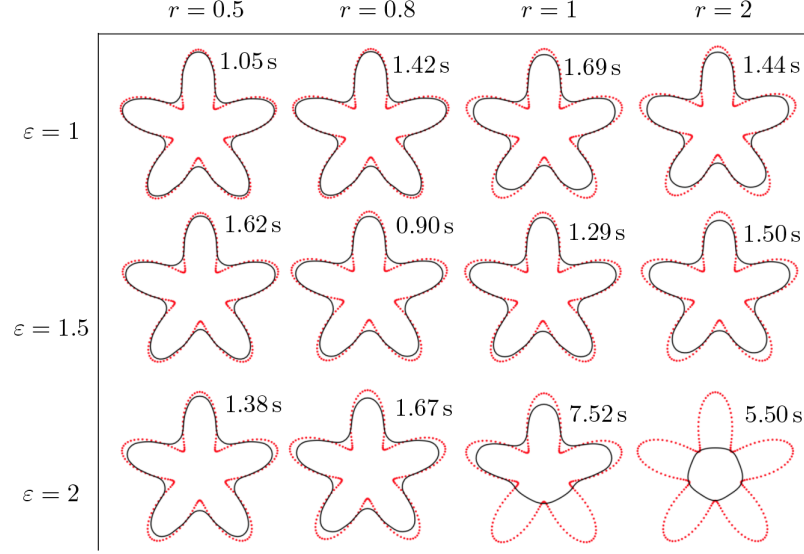


Figure 4.7: Results by ALM with different r and ε . For each column, from top to bottom, $\varepsilon = 1, 1.5, 2$; and for each row, from left to right, $r = 0.5, 0.8, 1, 2$. Increasing ε renders the curve less sharp and more convex. Increasing r induces a stronger diffusion effect on ϕ^n .

Equation 4.17 takes the following forms:

$$\Delta\phi^{n+1} = \begin{cases} -9 d |\mathbf{p}^n| / (8\sqrt{3}\pi \varepsilon^2 r) + \nabla \cdot (\mathbf{p}^n + \frac{\lambda^n}{r}) & \text{at } \mathbf{y}. \\ 9 d |\mathbf{p}^n| / (8\sqrt{3}\pi \varepsilon^2 r) + \nabla \cdot (\mathbf{p}^n + \frac{\lambda^n}{r}) & \text{at } \mathbf{z}. \end{cases} \quad (4.26)$$

The first terms in the right hand side of Equation 4.26 show that with a smaller value of ε , there are less number of points in B_ε , but the influence from d becomes stronger. With a larger value of ε , d affects more number of points in B_ε , but with a weaker influence. Varying values of r also modifies the effect of d , while the size of B_ε is not changed.

We also find that ε interacts with r and effectively modifies the shape of the level-set. Figure 4.7 shows the results for ALM using different combinations of r and ε , on the five-fold circle point cloud in Figure 4.1 (a). For a fixed r , increasing ε makes the approximated delta function smoother; consequently, narrow and elongated shapes are omitted, and the reconstructed curve becomes more convex. For a fixed ε , larger r loses more details, as discussed in subsection 4.1.4. The speed of convergence varies for different combinations

of r and ε . When the choices are reasonable, the algorithm converges fast within 2 seconds. When both r and ε are large, results are not as good, and the convergences are slow.

Another observation comes from Equation 4.20. For any point \mathbf{x} and $n \geq 0$, if the value

$$Q^n(\varepsilon, r) := \phi^n \pi |r \nabla \phi^n - \lambda^{n-1}| \varepsilon^2 - d\varepsilon + (\phi^n)^3 \pi |r \nabla \phi^n - \lambda^{n-1}|$$

is positive, then $\mathbf{p}^n(\mathbf{x}) = 0$, and d has no direct effect on Equation 4.18 at \mathbf{x} in the next iteration. Regarding $Q^n(\varepsilon, r)$ as a quadratic polynomial in terms of ε parameterized by r , the sign of $Q^n(\varepsilon, r)$ depends on the sign of $\phi^n(\mathbf{x})$ and the sign of its discriminant computed via:

$$\text{Disc } Q^n = d^2 - 4(\phi^n)^4 \pi^2 |r \nabla \phi^n - \lambda^{n-1}|^2.$$

The sign of $\phi^n(\mathbf{x})$ is related to the position of \mathbf{x} relative to the 0-level-set. The sign of $\text{Disc } Q^n$ is determined by comparing the length of a vector difference $r \nabla \phi^n - \lambda^{n-1}$ with the quantity $d/(4(\phi^n)^2 \pi)$. By the projection theorem, $|r \nabla \phi^n - \lambda^{n-1}|^2$ is bounded below by $\alpha^n := |\lambda^{n-1}|^2 - |\text{Proj}_{\nabla \phi^n} \lambda^{n-1}|^2 = |\lambda^{n-1}|^2 - |\lambda^{n-1} \cdot \nabla \phi^n|^2 / (|\lambda^{n-1}|^2 |\nabla \phi^n|^2)$, i.e., the squared residual of orthogonal projection of λ^{n-1} onto $\nabla \phi^n$; therefore, we can decide the sign of $\text{Disc } Q^n$ using r via the following cases:

1. When $\frac{d^2}{4(\phi^n)^4 \pi^2} < \alpha^n$, for any $r > 0$, $\text{Disc } Q^n < 0$.

2. When $\frac{d^2}{4(\phi^n)^4 \pi^2} \geq \alpha^n$:

(a) if $r > r_U^n$ or $r < r_L^n$, then $\text{Disc } Q^n < 0$;

(b) if $\max\{0, r_L^n\} \leq r \leq r_U^n$, then $\text{Disc } Q^n \geq 0$.

Here,

$$r_U^n = \frac{|\text{Proj}_{\nabla \phi^n} \lambda^{n-1}| + \sqrt{\frac{d^2}{4(\phi^n)^4 \pi^2} - \alpha^n}}{|\nabla \phi^n|} \quad \text{and} \quad r_L^n = \frac{|\text{Proj}_{\nabla \phi^n} \lambda^{n-1}| - \sqrt{\frac{d^2}{4(\phi^n)^4 \pi^2} - \alpha^n}}{|\nabla \phi^n|}.$$

When $\phi^n(\mathbf{x}) > 0$, Q^n concaves upwards and $Q^n(0, r) \geq 0$ for any r . If $\text{Disc } Q^n < 0$, Q^n is positive for all ε and d has no effect on level set evolution. If $\text{Disc } Q^n \geq 0$, Q^n is positive for ε outside the interval bounded by two roots of Q^n , i.e.,

$$0 < \varepsilon < \frac{d - \sqrt{\text{Disc } Q^n}}{2\phi^n\pi|r\nabla\phi^n - \lambda^{n-1}|} \quad \text{or} \quad \varepsilon > \frac{d + \sqrt{\text{Disc } Q^n}}{2\phi^n\pi|r\nabla\phi^n - \lambda^{n-1}|}.$$

When $\phi^n(\mathbf{x}) < 0$, Q^n concaves downwards, and $Q^n(0, r) \leq 0$ for any r . In this case, Q^n is never positive: either $\text{Disc } Q^n < 0$, i.e., no roots, or $\text{Disc } Q^n \geq 0$ but both roots are negative.

Notice that the bounds, r_L^n and r_U^n , are closely related to the ratio $d/(\phi^n)^2$, which contributes to the adaptive behavior of ALM. For example, for a point \mathbf{x} where $\phi^n(\mathbf{x}) > 0$, when $|\phi^n(\mathbf{x})|$ is close to 0 but $d(\mathbf{x}) \gg 0$, $r_L^n < 0$ and r_U^n becomes extremely large; thus, for a moderate value of r , d has a strong influence on the evolution of the level-set near \mathbf{x} and swiftly moves the curve towards the point cloud. For a point \mathbf{x} which is close to both \mathcal{D} and $\{\phi^n = 0\}$, the level-set evolution becomes more stringent about the minimization of the energy (Equation 4.10).

Figure 4.8 illustrates this effect, for the five-fold circle point cloud in Figure 4.1 (a) with $r = 2$ and $\varepsilon = 1$. Figure 4.8 shows (a) $\text{Disc } Q^n$, (b) r_U^n , (c) r_L^n , and (d) the region where d effects the level set evolution. The figures are for iterations $n = 2, 3, 4, 7, 8, 10, 11, 13$ and 38 (converged). The region inside $\{\phi^n = 0\}$ always experiences the influence of d , as described above. Figure 4.8 (a) shows that the region outside $\{\phi^n = 0\}$ is mostly blue indicating $\text{Disc } Q^n < 0$; hence, for almost every point outside the 0-level-set, as long as $r_L^n \leq r \leq r_U^n$, the landscape of d has strong effects on the evolution. In (b) and (c), observe that high values of r_U^n only concentrate near the 0-level-set while r_L^n remains relatively small in the whole domain; thus, the influence of d is strong near $\{\phi^n = 0\}$. (d) displays the white regions where d explicitly guides the level-set evolution and the black regions where d has no direct effect. These results show that, although ALM evolves the level-set

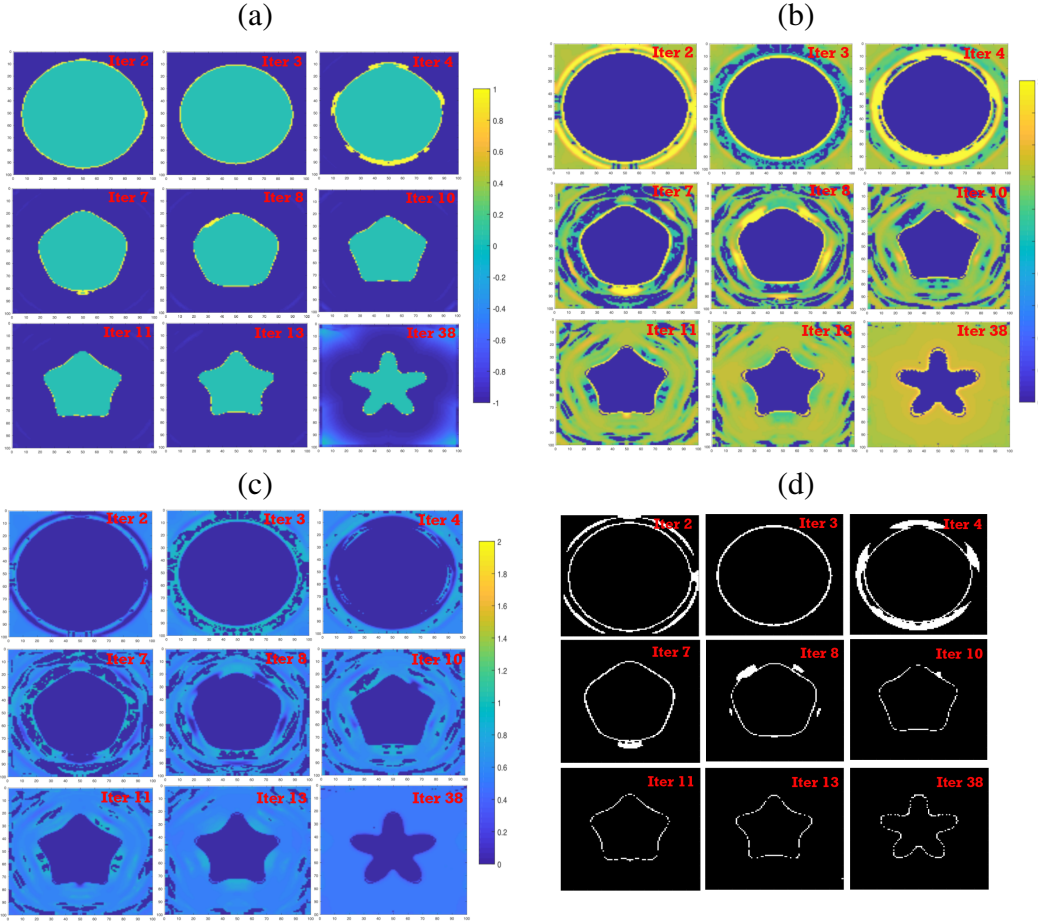


Figure 4.8: (a) Disc Q^n , (b) r_U^n , (c) r_L^n at certain iterations. (d) The region (in white) where d explicitly guides the level-set evolution by ALM. The distance function d refines the local structures and it is only active near $\{\phi^n = 0\}$. This partially explains the efficiency of ALM.

globally, it ignores the effects of d when evolving the regions far away from the level-sets; and it utilizes the values of d to refine the local structures for the regions of the level-sets close to \mathcal{D} .

4.3 Curvature-regularized Energy and Its Fast Optimizing Algorithms

From this section, we consider energy functionals with curvature constraints to enforce desired properties. One such curvature constraint is the squared mean curvature, κ^2 , such as Euler's elastica minimization model [289]. In addition to image inpainting, it has been applied to denoising [290], segmentation problem [291], and others. For any closed surface Γ in \mathbb{R}^3 , the bending energy,

$$\int_{\Gamma} \kappa^2 d\sigma,$$

where $d\sigma$ denotes the surface area element, is a conformal invariant [292], and it has a universal lower bound [293]: $\int_{\Gamma} \kappa^2 d\sigma \geq 4\pi$. Another curvature constraint we consider is the absolute mean curvature $|\kappa|$, i.e., $\int_{\Gamma} |\kappa| d\sigma$, which preserves sharp edges and corners in various cases, e.g., denoising [290, 294], and segmentation [295]. As a related work, in [282], graph cuts algorithm was explored for a functional with the absolute mean curvature term. In [296], a variation using a function which is sensitive to large curvature was considered. Other works used weighted mean curvature [297], principle curvature [298], Gaussian curvature [299], Menger curvature [300], and other high-order geometrical information, e.g., conformal factor [301] and elastic ratio [302].

Optimizing a curvature regularized functional is a non-convex and non-linear problem. Computation of such functional is particularly challenging. There are a number of different approaches to design a fast and efficient algorithm, e.g., multigrid method [303], graph-cut algorithm [282], homotopy method [304] and convex relaxation [305, 306], just to name a few. A semi-implicit scheme introduced in [280] simulates the curvature and surface diffusion motion of the interface. One of the major class of methods is based on *split-*

ting [307]. The common spirit of these methods is to cast the complicated primal problem into a series of more tamable subproblems, then to find the minimizer using alternative direction method. There are various strategies to obtain such decompositions from the optimization problem. One can derive the associated Euler-Lagrange equations, then apply operator splitting methods on the differential equations, e.g., Lie-Trotter method [308]. A new operator splitting algorithm was proposed for Euler’s elastica model for image smoothing in [290]. One can also introduce auxiliary variables and transform the primal problem into a constrained one, then obtain a series of subproblems by alternatively optimizing one variable at a time while keeping the others fixed; e.g., augmented Lagrangian method (ALM) [295, 287, 309, 310].

Specifically, we focus on a variational functional with a curvature constraint to reconstruct implicit surfaces from point cloud data, and explore fast algorithms to solve the associated non-convex, non-linear optimization problem proposed in [311]. The minimizing functional balances two terms, the Euclidean distance from the point cloud to the surface and the mean curvature of the surface. We show that the curvature term improves corner reconstruction and recovers non-convex features of the underlying shape of the point cloud data. To avoid dealing with the high-order PDEs resulting from the gradient descent approach, we introduce a semi-implicit method to solve an easier, but equivalent, problem derived by the operator splitting method (OSM). We also explore an ALM method recently proposed by Bae et al. [295], which reduces the number of parameters compared to other curvature regularized models. Our approaches work effectively for 2D/3D cases, as well as for noisy and sparse point cloud.

4.3.1 Curvature Regularized Surface Reconstruction Model

Let \mathcal{D} be the set of given point cloud data, $d(\mathbf{x}) = \inf_{\mathbf{y} \in \mathcal{D}} \{|\mathbf{x} - \mathbf{y}|\}$ measures the point-to-point-cloud distance, and $d\sigma$ is the area element. To reconstruct a surface from a given point

cloud data \mathcal{D} , we propose the following curvature-constrained minimal surface energy:

$$E_s(\Gamma) = \left(\int_{\Gamma} |d(\mathbf{x})|^s d\sigma \right)^{\frac{1}{s}} + \eta \left(\int_{\Gamma} |\kappa(\mathbf{x})|^s d\sigma \right)^{\frac{1}{s}}. \quad (4.27)$$

Here, κ is the mean curvature of Γ , and the exponent coefficient $s > 0$ is a constant integer. We explore the cases when $s = 1$ and $s = 2$. The first term, the surface integral of the distance from point cloud to the surface, signifies the fidelity of reconstruction. It moves the surface Γ closer toward the point cloud. The second term, which is the integral of the surface mean curvature along the reconstructed surface, is the regularization. This induces regularized geometric features for Γ independent to the point cloud location. Geometric features can include sharp corners, smooth corners, or straight segments, depending on the choice of s . The parameter $\eta > 0$ controls the influence of the curvature regularization. When $\eta = 0$, the model (Equation 4.27) degenerates to the minimal surface model proposed in [279].

Remark 4.3.1. *The $1/s$ power in Equation 4.27 comes from the original model (Equation 4.1). If one takes the distance function as the potential function of the point cloud, then the energy is an L_s norm of the potential on Γ . Based on this, we add the regularization term related to the mean curvature of Γ , and the $1/s$ power is used to keep the two terms in the same format. One may remove this power to get a simpler energy and still get the same minimizer when η is chosen appropriately.*

We employ implicit the surface representation [312] to rewrite the energy (Equation 4.27), and the level set function ϕ is defined such that Γ is its zero level set:

$$\phi(\mathbf{x}) \text{ is } \begin{cases} > 0, & \text{if } \mathbf{x} \text{ is outside of } \Gamma, \\ = 0, & \text{if } \mathbf{x} \text{ is on } \Gamma, \\ < 0, & \text{if } \mathbf{x} \text{ is inside } \Gamma. \end{cases}$$

Using ϕ , the functional $E_s(\Gamma)$ restricted to the surface Γ can be expressed as

$$E_s(\phi) = \left(\int_{\Omega} |d(\mathbf{x})|^s \delta(\phi) |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{s}} + \eta \left(\int_{\Omega} |\kappa(\mathbf{x})|^s \delta(\phi) |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{s}}, \quad (4.28)$$

where $\delta(\phi) = H'(\phi)$ is the Dirac Delta function with H being the Heaviside step function: $H(\phi) = 1$ if $\phi > 0$, and 0 otherwise. We use the smooth approximation [295] for practical computation

$$H_\varepsilon(\phi) = \frac{1}{2} + \frac{1}{\pi} \arctan \left(\frac{\phi}{\varepsilon} \right) \quad \text{and} \quad \delta_\varepsilon(\phi) = H'_\varepsilon(\phi) = \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)}, \quad (4.29)$$

with $\varepsilon > 0$, which is a constant controlling the smoothness. For any point \mathbf{x} on Γ , its mean curvature can be computed as, $\kappa(\mathbf{x}) = \nabla \cdot \left(\frac{\nabla \phi(\mathbf{x})}{|\nabla \phi(\mathbf{x})|} \right)$. Putting these together, we focus on the smoothed energy

$$E_{s,\varepsilon}(\phi) = \left(\int_{\Omega} |d(\mathbf{x})|^s \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{s}} + \eta \left(\int_{\Omega} \left| \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right|^s \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{s}} \quad (4.30)$$

and the reconstructed surface is defined as the minimizer of the energy (Equation 4.30), i.e.,

$$\Gamma = \{\mathbf{x} | \psi(\mathbf{x}) = 0\} \quad \text{for} \quad \psi = \arg \min_{\phi} E_{s,\varepsilon}(\phi).$$

Here ψ represents the optimal level set function.

4.3.2 Analytical Aspects

We consider the first variation of each term of the functional (Equation 4.27). The first variation of Equation 4.27 when $\eta = 0$ is [203]:

$$\frac{1}{s} \left(\int_{\Gamma} d^s(\mathbf{x}) d\sigma \right)^{1/s-1} (s d^{s-1} \nabla d \cdot \mathbf{n} + d^s \kappa), \quad (4.31)$$

which shows the interaction between the data-dependent driving force, d , and the shape geometric feature, κ . When Γ is close to the point cloud, i.e., d is small, the shape of Γ becomes flexible, i.e., κ can be large. When Γ is away from the point cloud, i.e., d is large, the shape of Γ becomes rigid, and κ must be small. For the regularization term, the effect of the mean-curvature κ of Γ is adjusted by the surface area. Notice that this term only focuses on the geometry of Γ , and the point cloud data has no influence. The first variation of the functional $(\int_{\Gamma} |\kappa(\mathbf{x})|^s d\sigma)^{1/s}$ can be derived as [296]:

$$\frac{1}{s} \left(\int_{\Gamma} \kappa(\mathbf{x})^s d\sigma \right)^{1/s-1} \times \begin{cases} \operatorname{div}_{\Gamma}(\delta(\kappa)\nabla_{\Gamma}\kappa) + \operatorname{sign}(\kappa)|W|^2 - \kappa|\kappa| & s = 1 \\ \operatorname{div}_{\Gamma}(|\kappa|^{s-2}s(s-1)\nabla_{\Gamma}\kappa) + s\kappa|\kappa|^{s-2}|W|^2 - \kappa|\kappa|^s & s \geq 2. \end{cases} \quad (4.32)$$

Here ∇_{Γ} is the tangent component of the gradient, and $\operatorname{div}_{\Gamma}$ is its dual operator; W is the Weingarten map of Γ , and $|W|$ equals the Gaussian curvature if Γ is a 2D surface. Compared to Equation 4.31, the first variation related to the regularization term (Equation 4.32) is more complicated. While the first variation (Equation 4.31) connects the distance and curvature, Equation 4.32 only depends on the geometric feature of the surface Γ .

When $s = 1$ in the model (Equation 4.27), we compare the cases with $\eta = 0$ and $\eta \neq 0$. We first note that κ of the minimizer can not be constantly zero, since there is no compact minimal surface. When $\eta = 0$, i.e., without curvature constraint, a minimizer of Equation 4.27 satisfies the necessary condition: $\nabla d \cdot \mathbf{n} + d\kappa = \nabla \cdot (d\mathbf{n}) = 0$; when $\eta \neq 0$, i.e., with the curvature constraint, the optimality condition becomes $\nabla \cdot (d\mathbf{n}) + \eta[\operatorname{div}_{\Gamma}(\delta(\kappa)\nabla_{\Gamma}\kappa) + \operatorname{sign}(\kappa)|W|^2 - |\kappa|\kappa] = 0$. On the open subset of the minimizer where $\kappa > 0$, this condition becomes $\nabla \cdot (d\mathbf{n}) = \eta[\kappa^2 - |W|^2]$, while on the region where $\kappa < 0$, it is $\nabla \cdot (d\mathbf{n}) = \eta[\kappa^2 + |W|^2]$. Hence, the curvature regularization modifies the distance weighted area of the minimizing surface depending on the local concavity/convexity and the Gaussian curvature. These modifications introduce more flexibility when fitting the

point cloud, and our experiments show that they can help to improve reconstruction results.

When $s = 2$, the first variation of the curvature regularization term $(\int_{\Gamma} \kappa^2 d\sigma)^{1/2}$ is

$$\left(\int_{\Gamma} \kappa(\mathbf{x})^2 d\sigma\right)^{-1/2} (\Delta_{\Gamma} \kappa + \kappa G^2 - \kappa^3/2), \quad (4.33)$$

where Δ_{Γ} is the Laplace-Beltrami operator, and G is the Gaussian curvature. Since Equation 4.33 contains $\Delta_{\Gamma} \kappa$, we expect to see that our model with $s = 2$ will be influenced by the locally averaged mean curvature, which leads to smoothing effects. Here we show this model's behavior in the following special case.

Proposition 4.3.1. *Suppose the point cloud is sampled from a smooth closed surface Γ with mean curvature κ and Gaussian curvature G satisfying:*

$$\kappa(G^2 - \kappa^2/2) = 0. \quad (4.34)$$

If the point cloud is sufficiently dense, i.e. the computed d is very close to the exact distance function, then Γ is a minimizer of Equation 4.27 only if it is a sphere of radius $\sqrt{2}$.

Proof. Since Γ passes through the point cloud, (Equation 4.31) degenerates to 0. Moreover, by Equation 4.34 together with Equation 4.32, the necessary condition for Γ being a minimizer is that $\Delta_{\Gamma} \kappa = 0$. Because Γ is closed, κ is a non-zero constant. By Equation 4.34, this implies that G is also constant; hence, we know that Γ can only be a sphere. Finally, since $G = 1/r^2$ and $\kappa = 1/r$ with r the radius of the sphere, we can solve for the radius of Γ , which is $\sqrt{2}$. \square

The curvature regularization term $(\int_{\Gamma} \kappa^2 d\sigma)^{1/2}$ inflates the membrane supported by the point cloud. Mylar balloon [313], which resembles slightly flattened sphere, satisfies the condition (Equation 4.34). Proposition 4.3.1 claims that, if the point cloud lies on a Mylar balloon, the minimizer of the functional (Equation 4.27) deviates from the underlying surface.

The following result shows a two dimensional example where the object is a circle, denoted by C_0 , which is centered at the origin with radius r_0 . It is fair to assume that a local minimizer of $E_s(\Gamma)$ is a circle, denoted by C , with the same center and radius close to r_0 . Denote the radius of C by r . Then we have the following proposition:

Proposition 4.3.2. *Under the setting of the above example, for $s = 1$, $r = r_0$ is a local minimizer of $E_1(C)$ for any η . For $s = 2$, $r = r_0$ is a local minimizer of $E_2(C)$ if $\eta \leq 2r_0$ and $r = (r_0 + \sqrt{r_0^2 + 12\eta})/6$ is a local minimizer if $\eta > 2r_0$.*

Proof. Note that for a circle with radius r and the same center as C_0 , $d = |r - r_0|$ and $\kappa = 1/r$. $E_s(C)$ can be written as

$$E_s(C) = (2\pi)^{\frac{1}{s}} \left(|r - r_0| r^{\frac{1}{s}} + \eta r^{\frac{1}{s}-1} \right).$$

For $s = 1$, $E_1(C) = (2\pi)^{\frac{1}{s}} \left(|r - r_0| r^{\frac{1}{s}} + \eta \right)$ of which C_0 is a local minimizer for any η .

For $s = 2$, $E_2(C) = (2\pi)^{\frac{1}{2}} \left(|r - r_0| r^{\frac{1}{2}} + \eta r^{-\frac{1}{2}} \right)$ whose subdifferential is

$$\partial E_2(C) = \begin{cases} (2\pi)^{\frac{1}{2}} \left(\frac{r_0}{2} r^{-\frac{1}{2}} - \frac{3}{2} r^{\frac{1}{2}} - \frac{\eta}{2} r^{-\frac{3}{2}} \right), & \text{for } r < r_0, \\ (2\pi)^{\frac{1}{2}} \left(\frac{3}{2} r^{\frac{1}{2}} - \frac{r_0}{2} r^{-\frac{1}{2}} - \frac{\eta}{2} r^{-\frac{3}{2}} \right), & \text{for } r > r_0. \end{cases}$$

For $r < r_0$, it can be easily shown that if $12\eta \leq r_0^2$ and $(r_0 + \sqrt{r_0^2 - 12\eta})/6 < r < r_0$, $\partial E_2(C) < 0$. If $12\eta > r_0^2$, $\partial E_2(C) < 0$ for any $r < r_0$. In other words, if $r < r_0$ and r is sufficiently close to r_0 , $\partial E_2(C) < 0$.

For $r > r_0$, if $\eta \leq 2r_0$, $\partial E_2(C) > 0$ and thus $r = r_0$ is a local minimizer of $E_2(C)$. If $\eta > 2r_0$, then $\partial E_2(C) < 0$ for $r_0 < r < (r_0 + \sqrt{r_0^2 + 12\eta})/6$ and $\partial E_2(C) > 0$ for $r > (r_0 + \sqrt{r_0^2 + 12\eta})/6$. Thus $r = (r_0 + \sqrt{r_0^2 + 12\eta})/6$ is a local minimizer of $E_2(C)$. \square

These properties show that the minimizer of the model (Equation 4.27) is not easy to

be analyzed even in a simple case such as a circle, and the results heavily depend on the combination of d and κ .

4.3.3 Operator Splitting Method (OSM)

One of our main challenges is that E_s in Equation 4.28 is highly nonlinear in terms of ϕ , and the corresponding Euler-Lagrange equation is a high-order nonlinear PDE. See Equation 4.31 and Equation 4.32 in subsection 4.3.2. To circumvent this difficulty, we propose a new operator splitting strategy, which leads to an equivalent differential equation system that is much easier to solve.

We follow the direction of gradient flow; however, we first decouple the data fidelity term and the curvature regularization term, then minimize the simplified functional via its gradient flow. For $s = 2$, using Equation 4.30, we rewrite the energy Equation 4.27 in the following equivalent form:

$$\begin{cases} \tilde{E}_{2,\varepsilon}(\phi) = \left(\int_{\Omega} d^2(\mathbf{x}) \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{2}} + \eta \left(\int_{\Omega} q^2(\mathbf{x}) \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x} \right)^{\frac{1}{2}}, \\ \text{with } q = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}, \end{cases} \quad (4.35)$$

with the notation $\delta_{\varepsilon}(\phi)$ as in Equation 4.29. We then compute the variation of $\tilde{E}_{2,\varepsilon}(\phi)$ with respect to ϕ [279]. For $\forall v \in H^2$, H^2 denoting the Sobolev space, we have

$$\begin{aligned} \left\langle \frac{\partial \tilde{E}_{2,\varepsilon}(\phi)}{\partial \phi}, v \right\rangle &= - \int_{\Omega} \frac{1}{2} \delta_{\varepsilon}(\phi) \left[\int_{\Omega} d^2(\mathbf{x}) \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x} \right]^{-1/2} \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] v d\mathbf{x} \\ &\quad - \eta \int_{\Omega} \frac{1}{2} \delta_{\varepsilon}(\phi) \left[\int_{\Omega} q^2(\mathbf{x}) \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x} \right]^{-1/2} \nabla \cdot \left[q^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] v d\mathbf{x}. \end{aligned}$$

If ψ is a minimizer of $\tilde{E}_{2,\varepsilon}$, it satisfies the optimality condition

$$\left\langle \frac{\partial \tilde{E}_{2,\varepsilon}(\psi)}{\partial \phi}, v \right\rangle = 0, \quad q - \nabla \cdot \frac{\nabla \psi}{|\nabla \psi|} = 0, \quad \forall v \in H^2. \quad (4.36)$$

To solve for ψ , we associate Equation 4.36 with the an initial value problem

$$\begin{cases} \frac{\partial \phi}{\partial t} = f(d, \phi) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] + \eta f(q, \phi) \nabla \cdot \left[q^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right], \\ \frac{\partial q}{\partial t} + \gamma(q - \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}) = 0, \end{cases} \quad (4.37)$$

with

$$f(d, \phi) = \frac{1}{2} \delta_\varepsilon(\phi) \left[\int_\Omega d^2(\mathbf{x}) \delta_\varepsilon(\phi) |\nabla \phi| d\mathbf{x} \right]^{-1/2}.$$

The steady state of Equation 4.37 is a minimizer of $\tilde{E}_{2,\varepsilon}$ in Equation 4.35. On the right hand side of the first equation in Equation 4.37, the two terms are of the same form, only differing by η , d and q . The first term is the driving velocity to minimize the squared distance from the surface to the given data. The second term is the driving velocity to minimize the squared curvature along the reconstructed surface. The parameter η controls the trade-off between these two terms.

We adopt the Lie type of operator splitting and refer the readers to [314] for a complete discussion of different splitting schemes. Given $\{\phi^k, q^k\}$ at the k -th step, we update $\{\phi^{k+1}, q^{k+1}\}$ in two fractional steps. In particular, for $k > 0$, we update the variables through $\{\phi^k, q^k\} \rightarrow \{\phi^{k+1/2}, q^{k+1/2}\} \rightarrow \{\phi^{k+1}, q^{k+1}\}$ as follows:

Fractional step I: Solve

$$\begin{cases} \frac{\partial \phi}{\partial t} = f(d, \phi) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] + \eta f(q, \phi) \nabla \cdot \left[q^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] \text{ on } \Omega \times [t^k, t^{k+1}], \\ \frac{\partial q}{\partial t} = 0 \text{ on } \Omega \times [t^k, t^{k+1}], \\ \phi(t^k) = \phi^k, q(t^k) = q^k \end{cases} \quad (4.38)$$

and set $\phi^{k+1/2} = \phi(t^{k+1}), q^{k+1/2} = q(t^{k+1})$.

Fractional step 2: Solve

$$\begin{cases} \frac{\partial \phi}{\partial t} = 0 \text{ on } \Omega \times [t^k, t^{k+1}] , \\ \frac{\partial q}{\partial t} + \gamma(q - \nabla \cdot \frac{\nabla \phi^{k+1/2}}{|\nabla \phi^{k+1/2}|}) = 0 \text{ on } \Omega \times [t^k, t^{k+1}] , \\ \phi(t^k) = \phi^{k+1/2}, q(t^k) = q^{k+1/2} . \end{cases} \quad (4.39)$$

and set $\phi^{k+1} = \phi(t^{k+1}), q^{k+1} = q(t^{k+1})$.

We have two subproblems Equation 4.38 and Equation 4.39 to address. There is no difficulty to solve Equation 4.39, since we have the closed form solution

$$q = e^{\gamma \Delta t} q^{k+1/2} + (1 - e^{\gamma \Delta t}) \nabla \cdot \frac{\nabla \phi^{k+1/2}}{|\nabla \phi^{k+1/2}|} .$$

To solve Equation 4.38 for $\phi^{k+1/2}$, the simplest way is to use the explicit scheme as the following:

$$\frac{\phi^{k+1/2} - \phi^k}{\Delta t} = f(d, \phi^k) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right] + \eta f(q^k, \phi^k) \nabla \cdot \left[(q^k)^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right] .$$

However, due to the stability consideration, one needs to choose a very small time step of order $O(h^2)$ where h is the spatial step size. To relax the time step constraint, for some $\alpha > 0$, we add $-\alpha \Delta \phi$ on both sides of Equation 4.38, as in [280], to get

$$\frac{\partial \phi}{\partial t} - \alpha \Delta \phi = -\alpha \Delta \phi + f(d, \phi) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] + \eta f(q, \phi) \nabla \cdot \left[q^2(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] . \quad (4.40)$$

We discretize (Equation 4.40) in time semi-implicitly:

$$\frac{\phi^{k+1/2} - \phi^k}{\Delta t} - \alpha \Delta \phi^{k+1/2} = -\alpha \Delta \phi^k + f(d, \phi^k) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right] + \eta f(q^k, \phi^k) \nabla \cdot \left[(q^k)^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right] .$$

We fix $\alpha = 1$ in this paper. This equation is a Laplacian equation of $\phi^{k+1/2}$ and can be

solved efficiently by fast Fourier transformation (FFT). The updating formula is summarized as

$$\begin{cases} \frac{\phi^{k+1} - \phi^k}{\Delta t} - \alpha \Delta \phi^{k+1} = -\alpha \Delta \phi^k + f(d, \phi^k) \nabla \cdot \left[d^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right] + \eta f(q^k, \phi^k) \nabla \cdot \left[(q^k)^2(\mathbf{x}) \frac{\nabla \phi^k}{|\nabla \phi^k|} \right], \\ q^{k+1} = e^{\gamma \Delta t} q^k + (1 - e^{\gamma \Delta t}) \nabla \cdot \frac{\nabla \phi^{k+1}}{|\nabla \phi^{k+1}|}. \end{cases} \quad (4.41)$$

To solve Equation 4.41, we need the initial condition (ϕ^0, q^0) . We choose ϕ^0 to be a signed distance function whose zero level set encloses all data. q^0 is assigned as $\nabla \cdot ((\nabla \phi^0)/|\nabla \phi^0|)$. The algorithm of OSM with $s = 2$ is stated in algorithm 7. In algorithm 7, the reinitialization is used to keep ϕ to be a signed-distance function near its zero level set. The details of reinitialization are discussed in subsection 4.3.5.

Initialization: d, ϕ^0, q^0 .

while not converge do
 | Update $\{\phi^{k+1}, q^{k+1}\}$ by solving Equation 4.41.
 | Reinitialize ϕ^{k+1} .
end
Output: ϕ^k .

Algorithm 7: Operator Splitting Method (OSM) for $s = 2$.

In the following, we give a brief derivation of OSM for $s = 1$. With $s = 1$, $E_{1,\varepsilon}$ has the same minimizer as

$$\begin{cases} \tilde{E}_{1,\varepsilon}(\phi) = \int_{\Omega} d(\mathbf{x}) \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x} + \eta \int_{\Omega} |q(\mathbf{x})| \delta_{\varepsilon}(\phi) |\nabla \phi| d\mathbf{x}, \\ \text{with } q = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}, \end{cases}$$

whose corresponding gradient flow initial value problem is

$$\begin{cases} \frac{\partial \phi}{\partial t} = \delta_{\varepsilon}(\phi) \nabla \cdot \left[d(\mathbf{x}) \frac{\nabla \phi}{|\nabla \phi|} \right] + \eta \delta_{\varepsilon}(\phi) \nabla \cdot \left[|q| \frac{\nabla \phi}{|\nabla \phi|} \right], \\ \frac{\partial q}{\partial t} + \gamma (q - \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}) = 0. \end{cases} \quad (4.42)$$

With the Lie type splitting in time and introducing the term $-\alpha\Delta\phi$ on both sides in the first equation of Equation 4.42, we get the updating formula

$$\begin{cases} \frac{\phi^{k+1} - \phi^k}{\Delta t} - \alpha\Delta\phi^{k+1} = -\alpha\Delta\phi^k + \delta_\varepsilon(\phi)\nabla \cdot \left[d(\mathbf{x}) \frac{\nabla\phi^k}{|\nabla\phi^k|} \right] + \eta\delta_\varepsilon(\phi)\nabla \cdot \left[|q^k| \frac{\nabla\phi^k}{|\nabla\phi^k|} \right], \\ q^{k+1} = e^{\gamma\Delta t} q^k + (1 - e^{\gamma\Delta t}) \nabla \cdot \frac{\nabla\phi^{k+1}}{|\nabla\phi^{k+1}|}. \end{cases} \quad (4.43)$$

We note that Equation 4.35 can be solved similarly by κ TV method proposed in [310] for image inpainting problem. One advantage of OSM in this paper is its simplicity and having less parameters: once the model parameter, i.e., η is fixed, there is only one parameter, the artificial time step, to tune. Numerical experiments show that there is a wide range of the time step we can choose.

4.3.4 Augmented Lagrangian Method (ALM)

We present another efficient algorithm to find the minimizer of Equation 4.30 with $s = 1$. The reason for only focusing on $s = 1$ in this case is that we can take advantage of the shrinkage operator. We first introduce three new variables: $\mathbf{p} = \nabla\phi$, $\mathbf{n} = \nabla\phi/|\nabla\phi|$ and $q = \nabla \cdot (\nabla\phi/|\nabla\phi|)$. Finding the minimizer of $E_{1,\varepsilon}$ becomes equivalent to solving

$$\min_{\phi, \mathbf{p}, \mathbf{n}, q} \int \varepsilon \frac{(d(\mathbf{x}) + \eta|q|)|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} d\mathbf{x} \quad \text{with} \quad \mathbf{p} = \nabla\phi, \quad \mathbf{n} = \nabla\phi/|\nabla\phi|, \quad q = \nabla \cdot (\nabla\phi/|\nabla\phi|).$$

This can be addressed via alternating direction method of multipliers by introducing

Lagrange multipliers $\lambda_1, \lambda_2, \lambda_3$. The associated Augmented Lagrangian functional is

$$\begin{aligned}
& \mathcal{L}(\phi, q, \mathbf{p}, \mathbf{n}, \lambda_1, \lambda_2, \lambda_3) \\
&= \int_{\Omega} \frac{\varepsilon(d + \eta|q|)|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} dx + \frac{r_1}{2} \int_{\Omega} |\mathbf{p} - \nabla \phi|^2 dx + \int_{\Omega} \lambda_1 \cdot (\mathbf{p} - \nabla \phi) dx \\
&+ \frac{r_2}{2} \int_{\Omega} (q - \nabla \cdot \mathbf{n})^2 dx + \int_{\Omega} \lambda_2 (q - \nabla \cdot \mathbf{n}) dx + \frac{r_3}{2} \int_{\Omega} \|\mathbf{p}|\mathbf{n} - \mathbf{p}\|^2 dx + \int_{\Omega} \lambda_3 \cdot (|\mathbf{p}|\mathbf{n} - \mathbf{p}) .
\end{aligned} \tag{4.44}$$

where $\mathbf{p}, \mathbf{n}, \lambda_1, \lambda_3$ are vectors, ϕ, q, λ_2 are scalars, r_1, r_2, r_3 are fixed constants. To find the saddle point of \mathcal{L} , we update each variable in an alternative manner. In each iteration, for each variable, we minimize the corresponding functional while keeping other variables fixed. After all variables are updated, we update Lagrange multipliers. This procedure is repeated until we achieve a steady state. In each iteration, we have four subproblems to minimize:

$$\mathcal{E}_1(\phi) = \int_{\Omega} \frac{\varepsilon(d + \eta|q|)|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} dx + \frac{r_1}{2} \int_{\Omega} |\mathbf{p} - \nabla \phi|^2 dx , + \int_{\Omega} \lambda_1 \cdot (\mathbf{p} - \nabla \phi) dx \tag{4.45}$$

$$\mathcal{E}_2(q) = \int_{\Omega} \frac{\varepsilon(d + \eta|q|)|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} dx + \frac{r_2}{2} \int_{\Omega} (q - \nabla \cdot \mathbf{n})^2 dx + \int_{\Omega} \lambda_2 (q - \nabla \cdot \mathbf{n}) dx , \tag{4.46}$$

$$\begin{aligned}
\mathcal{E}_3(\mathbf{p}) &= \int_{\Omega} \frac{\varepsilon(d + \eta|q|)|\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} dx + \frac{r_1}{2} \int_{\Omega} |\mathbf{p} - \nabla \phi|^2 dx + \int_{\Omega} \lambda_1 \cdot (\mathbf{p} - \nabla \phi) dx , \\
&+ \frac{r_3}{2} \int_{\Omega} \|\mathbf{p}|\mathbf{n} - \mathbf{p}\|^2 dx + \int_{\Omega} \lambda_3 \cdot (|\mathbf{p}|\mathbf{n} - \mathbf{p}) .
\end{aligned} \tag{4.47}$$

$$\begin{aligned}
\mathcal{E}_4(\mathbf{n}) &= \frac{r_2}{2} \int_{\Omega} (q - \nabla \cdot \mathbf{n})^2 dx + \int_{\Omega} \lambda_2 (q - \nabla \cdot \mathbf{n}) dx + \frac{r_3}{2} \int_{\Omega} \|\mathbf{p}|\mathbf{n} - \mathbf{p}\|^2 dx \\
&+ \int_{\Omega} \lambda_3 \cdot (|\mathbf{p}|\mathbf{n} - \mathbf{p}) .
\end{aligned} \tag{4.48}$$

After those four variables being updated correspondingly, Lagrange multipliers are updated as

$$\lambda_1 \leftarrow \lambda_1 + r_1(\mathbf{p} - \nabla \phi), \quad \lambda_2 \leftarrow \lambda_2 + r_2(q - \nabla \cdot \mathbf{n}), \quad \lambda_3 \leftarrow \lambda_3 + r_3(\mathbf{n}|\mathbf{p}| - \mathbf{p}) .$$

These subproblems can be solved efficiently as described in the following.

Subproblem of ϕ : For $\mathcal{E}_1(\phi)$ in Equation 4.45, the corresponding Euler-Lagrange equation is:

$$-r_1 \Delta \phi + \beta \phi = \beta \phi + (d + \eta |q|) \frac{2\varepsilon |\mathbf{p}| \phi}{\pi(\varepsilon^2 + \phi^2)^2} - \nabla \cdot (r_1 \mathbf{p} + \boldsymbol{\lambda}_1) ,$$

where $\beta > 0$ is a frozen coefficient. We discretize the time as follows

$$-r_1 \Delta \phi^{k+1} + \beta \phi^{k+1} = \beta \phi^k + (d + \eta |q^k|) \frac{2\varepsilon |\mathbf{p}^k| \phi^k}{\pi(\varepsilon^2 + (\phi^k)^2)^2} - \nabla \cdot (r_1 \mathbf{p}^k + \boldsymbol{\lambda}_1^k) . \quad (4.49)$$

This is the Laplacian equation of ϕ^{k+1} , and we efficiently solve it by FFT.

Subproblem of q : In Equation 4.46, $\mathcal{E}_2(\phi)$ can be written as:

$$\mathcal{E}_2(q) = \int_{\Omega} \frac{\eta \varepsilon |\mathbf{p}|}{\pi(\varepsilon^2 + \phi^2)} |q| + \frac{r_2}{2} \left(q - \left(\nabla \cdot \mathbf{n} - \frac{\lambda_2}{r_2} \right) \right)^2 dx + C ,$$

where C is independent of q . Then, the minimizer can be found via the shrinkage operator

$$\arg \min_q \mathcal{E}_2(q) = \max \left\{ 0, 1 - \frac{\eta \varepsilon |\mathbf{p}|}{r_2 \pi(\varepsilon^2 + \phi^2) |q^*|} \right\} q^* , \quad (4.50)$$

with $q^* = \nabla \cdot \mathbf{n} - \lambda_2/r_2$.

Subproblem of \mathbf{p} : In Equation 4.47, $\mathcal{E}_3(\phi)$ can be rewritten as

$$\begin{aligned} \mathcal{E}_3(\mathbf{p}) = & \int_{\Omega} \underbrace{\left[(d + \eta |q|) \frac{\varepsilon}{\pi(\varepsilon^2 + \phi^2)} + \boldsymbol{\lambda}_3 \cdot \mathbf{n} \right]}_{\omega} |\mathbf{p}| + \underbrace{\frac{r_1 + r_3(1 + |\mathbf{n}|^2)}{2}}_{\mu} \left| \mathbf{p} - \underbrace{\frac{\boldsymbol{\lambda}_3 + r_1 \nabla \phi - \boldsymbol{\lambda}_1}{r_1 + r_3(1 + |\mathbf{n}|^2)}}_{\mathbf{a}} \right|^2 \\ & - \int_{\Omega} \underbrace{r_3 \mathbf{n}}_{\nu} \cdot \mathbf{p} |\mathbf{p}| + \tilde{C} , \end{aligned}$$

where \tilde{C} is independent of \mathbf{p} . This $\mathcal{E}_3(\mathbf{p})$ can be simplified as

$$\mathcal{E}_3(\mathbf{p}) = \int_{\Omega} \omega |\mathbf{p}| + \frac{\mu}{2} |\mathbf{p} - \mathbf{a}|^2 - \boldsymbol{\nu} \cdot \mathbf{p} |\mathbf{p}| + \tilde{C} .$$

Following the idea of Theorem 2 in [295], we can minimize this energy efficiently.

Theorem 4.3.1. *Assume that $\mu > 2|\boldsymbol{\nu}|$. Let θ be the angle between \mathbf{a} and the minimum vector of $\mathcal{E}_3(\mathbf{p})$, and α is the angle between \mathbf{a} and $\boldsymbol{\nu}$. Then the following arguments hold:*

- if $\omega \geq \mu|\mathbf{a}|$, then $\arg \min_{\mathbf{p}} \mathcal{E}_3(\mathbf{p}) = \mathbf{0}$.

- if $\omega < \mu|\mathbf{a}|$:

1. if $\mathbf{a} = \boldsymbol{\nu} = \mathbf{0}$, then $\arg \min_{\mathbf{p}} \mathcal{E}_3(\mathbf{p}) = \begin{cases} \mathbf{0} , & \text{when } \omega \geq 0, \\ \text{any vector of length } -\omega/\mu , & \text{when } \omega < 0; \end{cases}$
2. if $\mathbf{a} \neq \mathbf{0}, \boldsymbol{\nu} = \mathbf{0}$, $\arg \min_{\mathbf{p}} \mathcal{E}_3(\mathbf{p}) = (1 - \frac{\omega}{\mu|\mathbf{a}|})\mathbf{a}$;
3. if $\mathbf{a} = \mathbf{0}, \boldsymbol{\nu} \neq \mathbf{0}$, $\arg \min_{\mathbf{p}} \mathcal{E}_3(\mathbf{p}) = \frac{\omega}{\mu-2|\boldsymbol{\nu}|} \frac{\boldsymbol{\nu}}{|\boldsymbol{\nu}|}$;
4. if $\mathbf{a} \neq \mathbf{0}, \boldsymbol{\nu} \neq \mathbf{0}$, the angles θ and α satisfy the equation:

$$\mu^2 |\mathbf{a}| \sin \theta + \mu |\boldsymbol{\nu}| |\mathbf{a}| \sin \theta \cos(\theta - \alpha) + \omega |\boldsymbol{\nu}| \sin(\theta - \alpha) + \mu |\mathbf{a}| |\boldsymbol{\nu}| \sin \alpha = 0 , \quad (4.51)$$

and $\arg \min_{\mathbf{p}} \mathcal{E}_3(\mathbf{p}) = \frac{[\mu(\mathbf{b} \cdot \mathbf{a}) - \omega] \mathbf{b}}{\mu + 2\boldsymbol{\nu} \cdot \mathbf{b}}$ with \mathbf{b} being a unit vector satisfying:

$$\mathbf{b} = \frac{1}{|\mathbf{a}|} \begin{bmatrix} \cos \tilde{\theta} & -\sin \tilde{\theta} \\ \sin \tilde{\theta} & \cos \tilde{\theta} \end{bmatrix} \mathbf{a} ,$$

and $\tilde{\theta} = \theta$ if $\det[\boldsymbol{\nu} \ \mathbf{a}] \geq 0$, $\tilde{\theta} = -\theta$ if $\det[\boldsymbol{\nu} \ \mathbf{a}] < 0$. Here $[\boldsymbol{\nu} \ \mathbf{a}]$ denotes the 2×2 matrix with the vector $\boldsymbol{\nu}$ and \mathbf{a} being the first and second column respectively.

Note that the condition in Theorem 4.3.1 is always satisfied since $\mu = \frac{r_1 + r_3(1 + |\mathbf{n}|^2)}{2}$, $\boldsymbol{\nu} = r_3 \mathbf{n}$ and $r_3(1 + |\mathbf{n}|^2) \geq 2r_3|\mathbf{n}|$ for any \mathbf{n} . From Equation 4.51, θ is solved by Newton's

method.

Subproblem of \mathbf{n} . For $\mathcal{E}_4(\phi)$ in Equation 4.48, the Euler-Lagrange equation is:

$$-r_2 \nabla(\nabla \cdot \mathbf{n}) + D\mathbf{n} = (D - r_3|\mathbf{p}|^2)\mathbf{n} - \nabla(r_2 q + \lambda_2) - (\boldsymbol{\lambda}_3 - r_3 \mathbf{p})|\mathbf{p}|, \quad (4.52)$$

where $D = \max_{x \in \Omega}(r_3|\mathbf{p}|^2 + \beta_2)$ and β_2 is a small positive number. We discretize in time as

$$-r_2 \nabla(\nabla \cdot \mathbf{n}^{n+1}) + D\mathbf{n}^{n+1} = (D - r_3|\mathbf{p}^n|^2)\mathbf{n}^n - \nabla(r_2 q^n + \lambda_2^n) - (\boldsymbol{\lambda}_3^n - r_3 \mathbf{p}^n)|\mathbf{p}^n|. \quad (4.53)$$

which can be solved efficiently by FFT.

For the initial condition, we use the same ϕ^0 and q^0 as that in OSM. For other variables, we use $\mathbf{p}^0 = \nabla \phi^0$, $\mathbf{n}^0 = \mathbf{p}^0/|\mathbf{p}^0|$, $\boldsymbol{\lambda}_1^0 = \boldsymbol{\lambda}_3^0 = \mathbf{0}$, $\lambda_2^0 = 0$.

The outline of augmented Lagrangian is summarized in algorithm 8.

4.3.5 Implementation Details

For a rectangular domain $\Omega = [0, M] \times [0, N] \in \mathbb{R}^2$ with M, N being positive integers, we discretize it by a Cartesian grid with $\Delta x = \Delta y = 1$. For any function u (resp. $\mathbf{v} = (v^1, v^2)^T$) defined on Ω , we use $u_{i,j}$ (resp. $\mathbf{v}_{i,j} = (v_{i,j}^1, v_{i,j}^2)^T$) to denote $u(i\Delta x, j\Delta y)$ (resp. $\mathbf{v}(i\Delta x, j\Delta y)^T$) for $0 \leq i \leq M, 0 \leq j \leq N$. Denote the standard forward and backward difference as

$$\begin{aligned} \partial_1^- u_{i,j} &= \begin{cases} u_{i,j} - u_{i-1,j}, & 1 < i \leq M; \\ u_{1,j} - u_{M,j}, & i = 1. \end{cases} & \partial_1^+ u_{i,j} &= \begin{cases} u_{i+1,j} - u_{i,j}, & 1 \leq i < M-1; \\ u_{1,j} - u_{M,j}, & i = M. \end{cases} \\ \partial_2^- u_{i,j} &= \begin{cases} u_{i,j} - u_{i,j-1}, & 1 < j \leq N; \\ u_{i,1} - u_{i,N}, & j = 1. \end{cases} & \partial_2^+ u_{i,j} &= \begin{cases} u_{i,j+1} - u_{i,j}, & 1 \leq j < N-1; \\ u_{i,1} - u_{i,N}, & j = N. \end{cases} \end{aligned}$$

Initialization: $d, \phi^0, q^0, \mathbf{p}^0, \mathbf{n}^0, \boldsymbol{\lambda}_1^0, \lambda_2^0, \boldsymbol{\lambda}_3^0$.

while not converge do

Update variables

Update $\phi^{k+1} = \arg \min_{\phi} \mathcal{L}(\phi, q^k, \mathbf{p}^k, \mathbf{n}^k, \boldsymbol{\lambda}_1^k, \lambda_2^k, \boldsymbol{\lambda}_3^k)$ by solving (Equation 4.49).

Update $q^{k+1} = \arg \min_q \mathcal{L}(\phi^k, q, \mathbf{p}^k, \mathbf{n}^k, \boldsymbol{\lambda}_1^k, \lambda_2^k, \boldsymbol{\lambda}_3^k)$ by solving (Equation 4.50).

Update $\mathbf{p}^{k+1} = \arg \min_{\mathbf{p}} \mathcal{L}(\phi^k, q^k, \mathbf{p}, \mathbf{n}^k, \boldsymbol{\lambda}_1^k, \lambda_2^k, \boldsymbol{\lambda}_3^k)$ according to Theorem Theorem 4.3.1.

Update $\mathbf{n}^{k+1} = \arg \min_{\mathbf{n}} \mathcal{L}(\phi^k, q^k, \mathbf{p}^k, \mathbf{n}, \boldsymbol{\lambda}_1^k, \lambda_2^k, \boldsymbol{\lambda}_3^k)$ by solving (Equation 4.53).

Reinitialize ϕ^{k+1} .

Update Lagrange multipliers:

$$\begin{aligned}\boldsymbol{\lambda}_1^{k+1} &= \boldsymbol{\lambda}_1^k + r_1(\mathbf{p}^{k+1} - \nabla \phi^{k+1}), \\ \lambda_2^{k+1} &= \lambda_2^k + r_2(q^{k+1} - \nabla \cdot \mathbf{n}^{k+1}), \\ \boldsymbol{\lambda}_3^{k+1} &= \boldsymbol{\lambda}_3^k + r_3(\mathbf{n}^{k+1} |\mathbf{p}^{k+1}| - \mathbf{p}^{k+1}).\end{aligned}$$

end

Output: ϕ^k .

Algorithm 8: Augmented Lagrangian Method (ALM) for $s = 1$.

The gradient, divergence and the Laplacian operators are approximated as follows:

$$\begin{aligned}\nabla u_{i,j} &= ((\partial_1^- u_{i,j} + \partial_1^+ u_{i,j})/2, (\partial_2^- u_{i,j} + \partial_2^+ u_{i,j})/2) , \\ \nabla \cdot \mathbf{v}_{i,j} &= (\partial_1^+ v_{i,j}^1 + \partial_1^- v_{i,j}^1)/2 + (\partial_2^+ v_{i,j}^2 + \partial_2^- v_{i,j}^2)/2 , \\ \Delta u_{i,j} &= \partial_1^+ u_{i,j} - \partial_1^- u_{i,j} + \partial_2^+ u_{i,j} - \partial_2^- u_{i,j} .\end{aligned}$$

Denote the discrete Fourier transform and its inverse by \mathcal{F} and \mathcal{F}^{-1} , respectively. For a function u , we have

$$\mathcal{F}(u)(i \pm 1, j) = e^{\pm 2\pi\sqrt{-1}(i-1)/M} \mathcal{F}(u)(i, j), \quad \mathcal{F}(u)(i, j \pm 1) = e^{\pm 2\pi\sqrt{-1}(j-1)/N} \mathcal{F}(u)(i, j) ,$$

which gives rise to

$$\mathcal{F}(\partial_1^- u)(i, j) = (1 - e^{-2\pi\sqrt{-1}(i-1)/M}) \mathcal{F}(u)(i, j) ,$$

and $\mathcal{F}(\partial_1^+ u)(i, j)$, $\mathcal{F}(\partial_2^- u)(i, j)$ and $\mathcal{F}(\partial_2^+ u)(i, j)$ can be computed similarly. Both OSM and ALM use FFT to enhance the computational efficiency. The first equation of Equation 4.43 and the first equation of Equation 4.49 belong to the same class:

$$-a\Delta u + bu = c , \tag{4.54}$$

for a function u with constants $a, b > 0$ and constant c . Using the Fourier transform, we have

$$\mathcal{F}(\Delta u)(i, j) = [2 \cos(\pi\sqrt{-1}(i-1)/M) + 2 \cos(\pi\sqrt{-1}(j-1)/N) - 4] \mathcal{F}u(i, j) .$$

Then Equation 4.54 can be solved by

$$u = \mathcal{F}^{-1} \left(\frac{\mathcal{F}(c)}{b - a (2 \cos(\pi \sqrt{-1}(i-1)/M) + 2 \cos(\pi \sqrt{-1}(j-1)/N) - 4)} \right).$$

The equation (Equation 4.52) is in the form of

$$-a \nabla (\nabla \cdot \mathbf{v}) + b \mathbf{v} = \mathbf{c}, \quad (4.55)$$

for some vector valued function $\mathbf{v} = (v^1, v^2)^T$ where a, b are constant positive scalars and $\mathbf{c} = (c^1, c^2)^T$ is a vector valued constant. After the distretization, Equation 4.55 can be written as

$$\begin{cases} -a \nabla (\partial_1^+ \partial_1^- v^1 + \partial_1^+ \partial_2^- v^2) + b v^1 = c^1, \\ -a \nabla (\partial_2^+ \partial_1^- v^1 + \partial_2^+ \partial_2^- v^2) + b v^2 = c^2. \end{cases} \quad (4.56)$$

Applying the discrete Fourier transform on both sides of Equation 4.56, we get

$$A \begin{bmatrix} \mathcal{F}(v^1)(i, j) \\ \mathcal{F}(v^2)(i, j) \end{bmatrix} = \begin{bmatrix} \mathcal{F}(c^1)(i, j) \\ \mathcal{F}(c^2)(i, j) \end{bmatrix}, \quad (4.57)$$

with

$$A = \begin{bmatrix} b - a(e^{\sqrt{-1}(i-1)/M} - 1)(1 - e^{-\sqrt{-1}(i-1)/M}) & -a(e^{\sqrt{-1}(i-1)/M} - 1)(1 - e^{-\sqrt{-1}(j-1)/N}) \\ -a(e^{\sqrt{-1}(j-1)/N} - 1)(1 - e^{-\sqrt{-1}(i-1)/M}) & b - a(e^{\sqrt{-1}(j-1)/N} - 1)(1 - e^{-\sqrt{-1}(j-1)/N}) \end{bmatrix}.$$

Hence, \mathbf{v} can be computed by first solving Equation 4.57 for $(\mathcal{F}(v^1), \mathcal{F}(v^2))^T$ and then apply inverse Fourier transform.

For any $\mathbf{x} \in \Omega$, $d(\mathbf{x})$ is the distance from \mathbf{x} to the collection of the point cloud \mathcal{D} , and

it can be computed by solving the Eikonal equation

$$\begin{cases} |\nabla d| = 1, \\ d(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \mathcal{D}. \end{cases} \quad (4.58)$$

The simplest monotonic scheme to discretize Equation 4.58 is the Lax-Friedrich scheme which leads to the updating formula [288]:

$$d_{i,j}^{n+1} = \frac{1}{2} \left(1 - |\nabla d_{i,j}^n| + \frac{d_{i+1,j}^n + d_{i-1,j}^n}{2} + \frac{d_{i,j+1}^n + d_{i,j-1}^n}{2} \right).$$

This is updated with a fast sweeping method.

To make our algorithm robust, when updating ϕ , we reinitialize the level set to be a signed distance function via solving

$$\phi_\tau + \text{sign}(\phi)(1 - |\nabla \phi|) = 0.$$

In practice, after each iteration, we only solve this PDE for a few iterations. For three dimensional space, we use a simple extension of the two dimensional case.

Remark 4.3.2. *In three-dimensional surface reconstruction problems, the point cloud can be large and the computer memory is limited. One can consider a narrow tube which encloses the point cloud, and assume the reconstructed surface lies inside this tube during the evolution. Adopting the local level set method such as [315], only the values of the level set function on grid points inside the tube need to be stored. ALM and OSM can be applied under the local level set method framework except for solving the Laplace equations. Under this framework, one can derive a corresponding linear system for each Laplace equation which can be solved efficiently by the conjugate gradient method.*

4.4 Numerical Results and Comparisons

We present numerical results of the proposed model (Equation 4.27). Without specification, when OSM is used, we refer to algorithm 7 for model (Equation 4.35) with $\gamma = 10$, $\alpha = 1$ and $\Delta t = 50$; when ALM is used, we refer to Algorithm algorithm 8 for model (Equation 4.44) with $\beta = 0.1$. For a fixed η , OSM only has one (the time step) parameter, whereas ALM has three parameters (r_1, r_2, r_3) . We use domain $[0, 100]^2$ for two dimensional problems and $[0, 50]^3$ for three dimensional problems. In all examples, $\varepsilon = 1$ is used.

In this section, we first consider the effect of parameters for ALM. Second, we compare the performance of ALM and OSM. We find that using OSM with $s = 2$ gives the best results. We conclude this section by several examples to further explore the performance of OSM when $s = 2$.

4.4.1 Choice of Parameters for ALM Method

In the case of ALM, the choice and combinations of the parameters are delicate. When r_1 or r_2 is increased, the reconstruction becomes closer to the point cloud. In Figure 4.9, we fix $r_2 = 10$, $r_3 = 3$, and $\eta = 2$, and let r_1 vary. With increased r_1 , ALM renders the curve closer to the point cloud. In Figure 4.10, we fix $r_1 = 10$, $r_3 = 3$, and $\eta = 2$, and let r_2 vary; larger r_2 induces better reconstruction. For this example, r_3 has little influence, yet, with a large r_3 , the results may become unstable or divergent.

In Figure 4.11, we fix $r_1 = 15$, $r_2 = 10$, and $r_3 = 3$, and increase η . With more influence on the curvature, as η is increased from 0 to 1, the indent on the rectangle is better reconstructed. When we increase η from 1 to 5, we see that, the indent is preserved, yet the tip of the wedge does not extend inward as much as in the case where $\eta = 1$. This is because a large value of η encourages both small mean curvature and short curve length. We find that increasing η also helps to avoid oscillation during the iteration. In Figure 4.11,

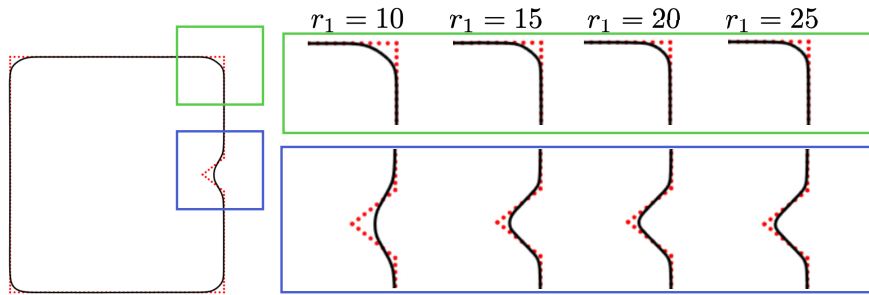


Figure 4.9: Effect of r_1 in ALM. For fixed $r_2 = 10$, $r_3 = 3$, and $\eta = 2$, increasing r_1 induces better reconstruction on the concave part.

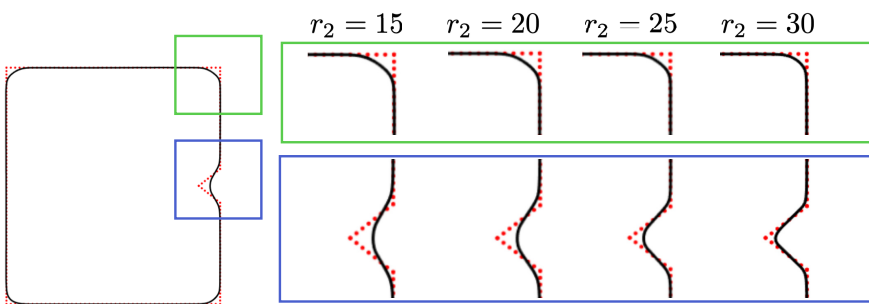


Figure 4.10: Effect of r_2 in ALM. For fixed $r_1 = 10$, $r_3 = 3$, and $\eta = 2$, increasing r_2 induces better reconstruction on the concave part.

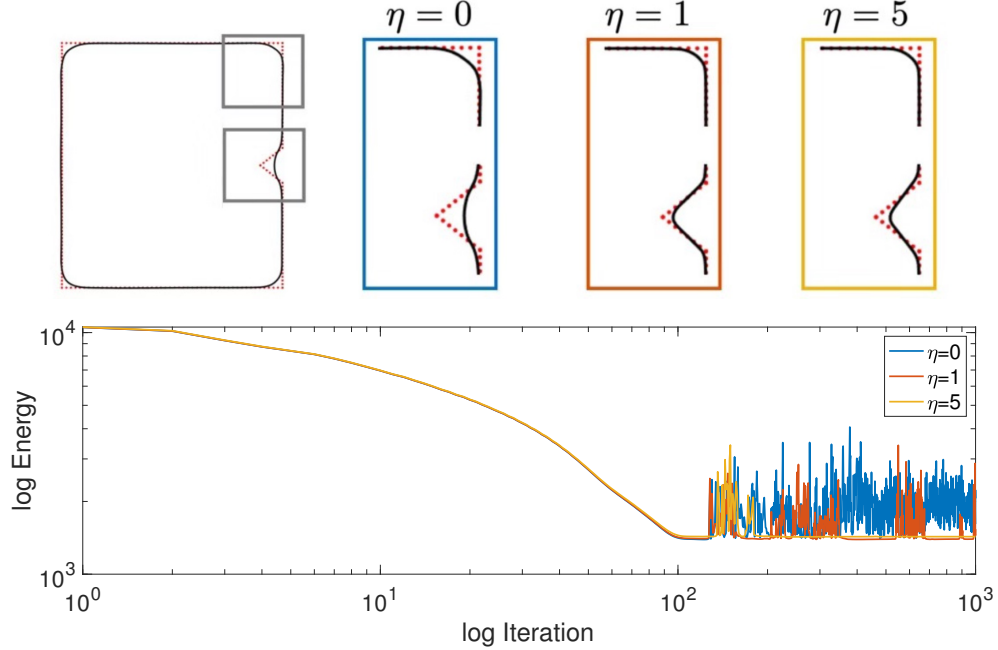


Figure 4.11: Effect of η in ALM. Here $r_1 = 15$, $r_2 = 10$, and $r_3 = 3$ are fixed. Increasing η induces reconstruction of the concave wedge. Although in cases, the energy curves are identical before the 100-th iteration, larger η suppresses the oscillation of the energy curve: yellow line ($\eta = 5$) is more stable compared to red ($\eta = 1$) or blue ($\eta = 0$).

we plot the energy curves corresponding to the these cases. Before the 100-th iteration, these curves are indistinguishable; however, after the 100-th iteration, larger values of η suppress the oscillation of the energy curve, which gives a more stable convergence.

Figure 4.12 shows the robustness against noise. The point cloud is sampled from a circle in $\Omega = [0, 200]^2$, and Gaussian noise with standard deviation 2 is added to the data. Using ALM with $r_1 = 15, r_2 = 10, r_3 = 3$, η is varied from 0, 1, to 10. Figure 4.12 (a) shows similar performances, while the zoomed-in results in Figure 4.12 (b) show that the larger the η the smoother the result becomes.

4.4.2 Comparison between OSM and ALM

Figure 4.13 (a) shows the energy convergence comparison between OSM (with $s = 2$) and ALM ($s = 1$). In ALM, $r_1 = 15, r_2 = 10, r_3 = 3$ is used. Convergence to the steady state is faster for OSM. The reconstructed curves are shown in Figure 4.13 (b) and (c). ALM

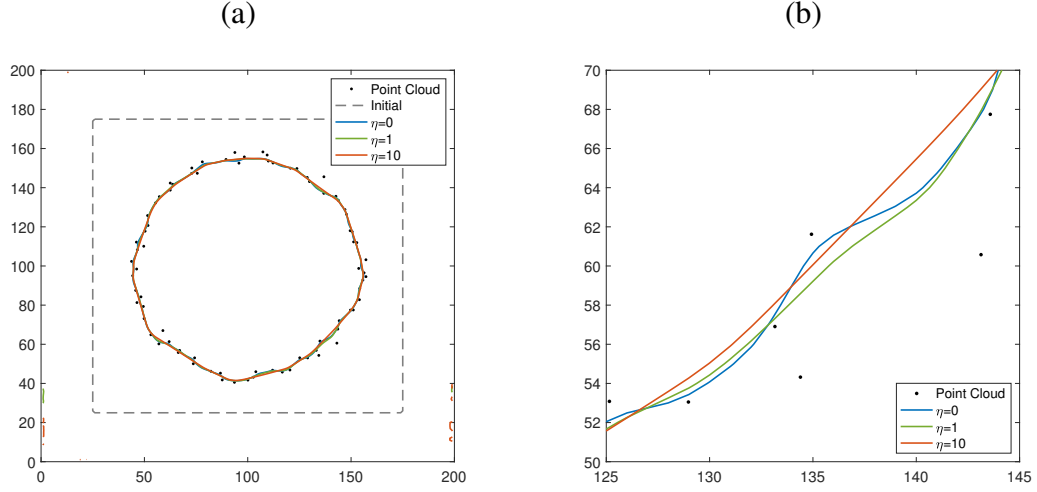


Figure 4.12: (a) Results by ALM with noisy data and $r_1 = 15, r_2 = 10, r_3 = 3$. (b) shows a zoom-in of the right-bottom of (a). The noise is additive Gaussian with standard deviation 2. As η increases, the curve becomes less oscillatory.

with $s = 1$ prefers to shorten the length, since it allows sharp corners. There is a balance between the distance term and the regularization term in the functional. OSM performs better in preserving corners, while the results extrude out a little bit at all corners. In this case of $s = 2$, the reconstruction is more circular, since sharp corners are not allowed.

Using OSM, we fix $s = 2$ for the distance term in Equation 4.27 and explore the difference between using (I) no curvature term $\eta = 0$, (II) L_1 norm of the mean curvature, and (III) L_2 norm of the mean curvature. Figure 4.14 (a) shows the comparison between

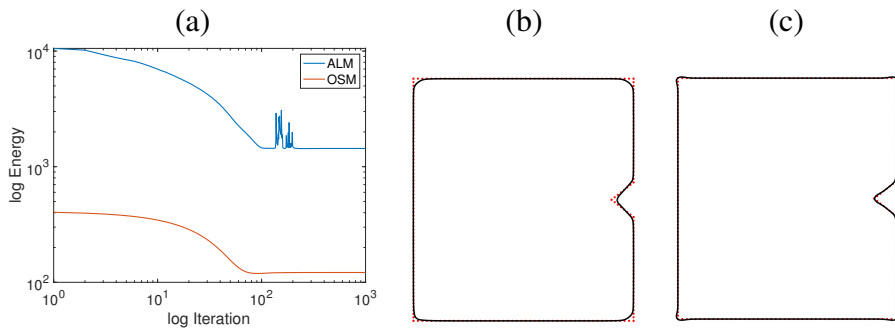
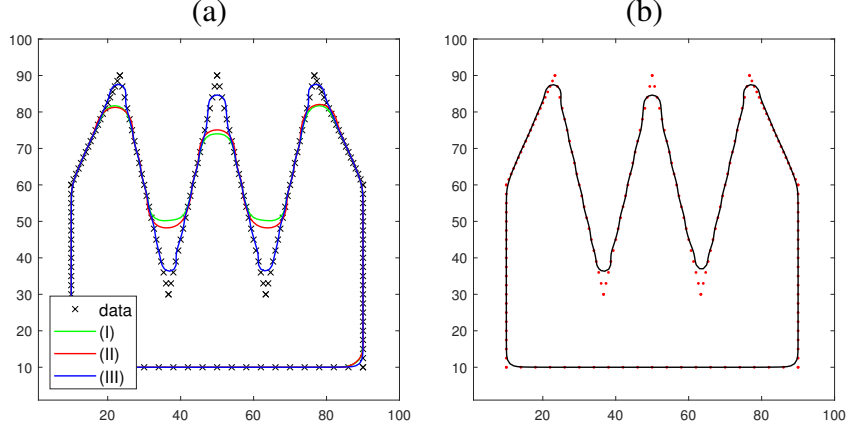


Figure 4.13: With $\eta = 2$, comparison between OSM with $s = 2$ and ALM. In ALM, $r_1 = 15, r_2 = 10, r_3 = 3$ is used. Convergence to the steady state is faster for OSM. The reconstructed curves are shown in (b) for ALM and in (c) for OSM: ALM may shorten the curve, while OSM can extrude a corner to make a circular reconstruction.

Figure 4.14: (a) By OSM with $s = 2$ for the distance term in (Equation 4.27), the comparison between (I) $\eta = 0$ (green curve), (II) $s = 1$ (red curve), and (III) $s = 2$ (blue curve) for the curvature term. (b) OSM with $\eta = 2.5$ for $s = 2$ in the model (Equation 4.35). This is the blue curve in (a). OSM using $s = 2$ gives the best result in terms of capturing the structure of the underlying surface more accurately.



$\eta = 0$ (green curve), $s = 1$ (red curve), and $s = 2$ (blue curve). The blue curve (OSM with $s = 2$) in (a) is presented separately in Figure 4.14 (b). OSM using $s = 2$ gives the best result capturing the structure of the underlying surface more accurately. The rest of the numerical experiments use OSM with $s = 2$, which gives more stable results with less number of parameters and faster convergence with smaller minimized energies.

4.4.3 Effect of curvature constraint: OSM with $s = 2$

Figure 4.15 shows the comparison between the algorithm from [274] (the first row), $\eta = 0$ (the second row), and OSM with $\eta > 0$ (the third row) for different surfaces. In the algorithm from [274], $r_1 = r_2 = 8, r_3 = r_4 = 3$ are used and the reinitialization is used to post-process the surface. For the boomerang shape in column (d), the surface constructed by [274] first shrinks to a point and then disappears. From this comparison, OSM with the curvature regularization provides the best results. The performance of the algorithm from [274] is similar to that of OSM without curvature regularization.

With the curvature term, the shape of the underlying surface is better captured in the third row. The interior triangle shape of Figure 4.15 (a), the first column, is better captured

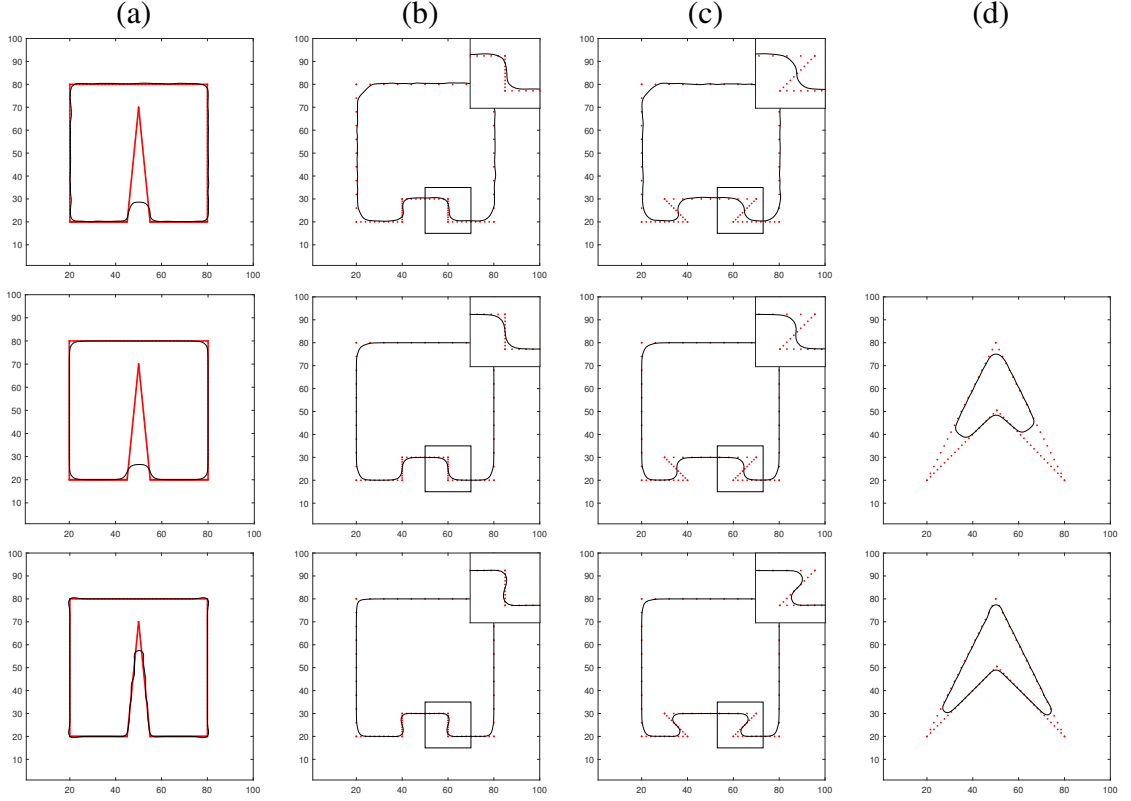


Figure 4.15: Comparison between the algorithm from [274] and OSM with or without curvature constraints: The first row results are by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. The second row results are by OSM without any curvature term, $\eta = 0$. The third row results are by OSM with curvature constraint ($s = 2$): (a) $\eta = 3$, (b) $\eta = 2$, (c) $\eta = 1$, and (d) $\eta = 2$. The shape of the underlying surface are more accurately captured using our proposed model with the curvature constraint.

with the curvature term. Without the curvature term, the reconstructed curve does not move further inside, since the prominent part has attained the balance between the curve length and its distance to the two sides of the triangle. Notice that the prominent part in the second row has non-zero curvature. With a positive η , this balance is broken and the prominent part will further move towards the upper vertex of the triangle. Also, for the cases with sharp corners in Figure Figure 4.15 (c)-(d), our model with the curvature term improves the results and recover the underlying shape better.

As η increases, different effect can be shown. See Figure 4.16. As one increases η , the two sharp corners are recovered better. However, if η is too large, like 3 and 4 in this example, the corners get more circular. As η , the weight of curvature term, gets larger, the

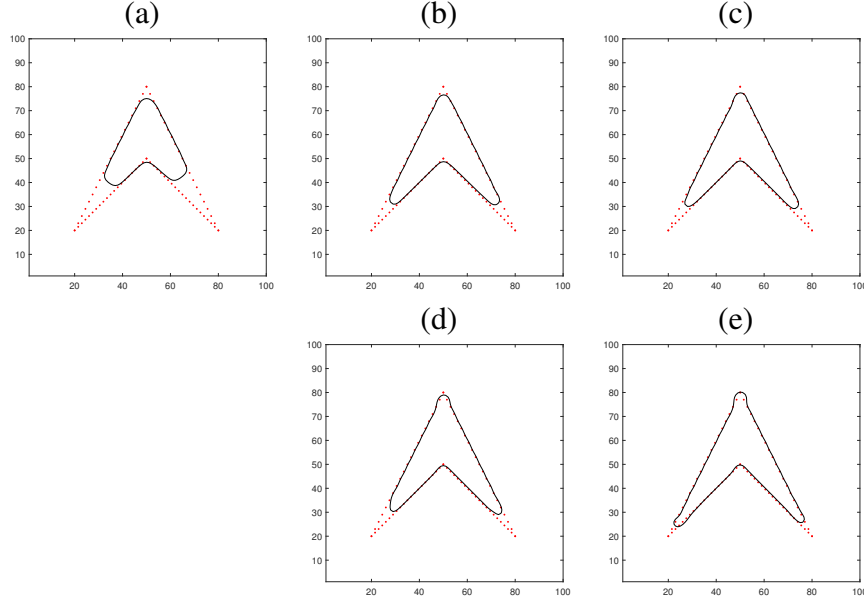


Figure 4.16: Effect of η in OSM. (a) $\eta = 0$. (b) $\eta = 1$. (c) $\eta = 2$. (d) $\eta = 3$. (e) $\eta = 4$. As η increases, the two sharp corners are recovered better. As η gets larger, the corners get more circular to avoid large curvature.

reconstructed curve becomes even more circular to avoid large curvature.

Figure 4.17 shows results when the given point cloud are sparse. The point cloud for (a) and (b) is a boomerang shape and that for (c) and (d) is a sparse square with an indent at the bottom. With sparse boomerang data, just using the distance term ($\eta = 0$) can recover very limited part of the given point cloud; see Figure 4.17 (a). With $\eta = 4$, We recover the general shape of boomerang in Figure 4.17 (b). For the sparse square shape, only four corners and one point exist on each side in Figure 4.17 (c)-(d). While for $\eta = 0$, the bottom part of the recovered shape is smooth. With $\eta = 2$, the rectangle shape is clearly recovered showing corners in the bottom area in Figure 4.17 (d). Figure 4.18 shows the case where even less number of points are given. See Figure 4.18 (a). Only two points around each corner are given. Figure 4.18 (b) and (c) show results with $\eta = 1$ and $\eta = 1.5$, respectively. Even with extremely sparse data, curvature constraint model can reconstruct the corners well. We observe that when the given point cloud is non-uniform, or data are missing in some region, our algorithm yields results with straight edges between distant points and

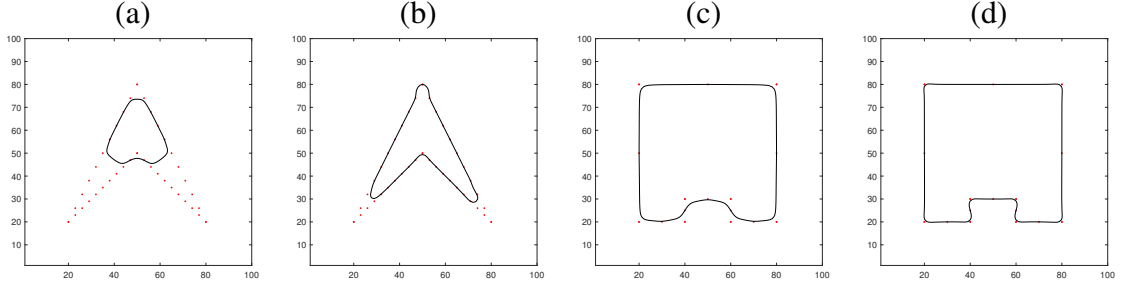


Figure 4.17: By OSM, sparse data results with or without curvature constraints. (a) $\eta = 0$, and (b) $\eta = 4$ for a point cloud sampled from a Boomerang shape. (c) $\eta = 0$ and (d) $\eta = 2$, for a sparse square shape where only four corners and one point on each side are given. For both examples, with curvature constraint, the recovery is more accurate and sharper.

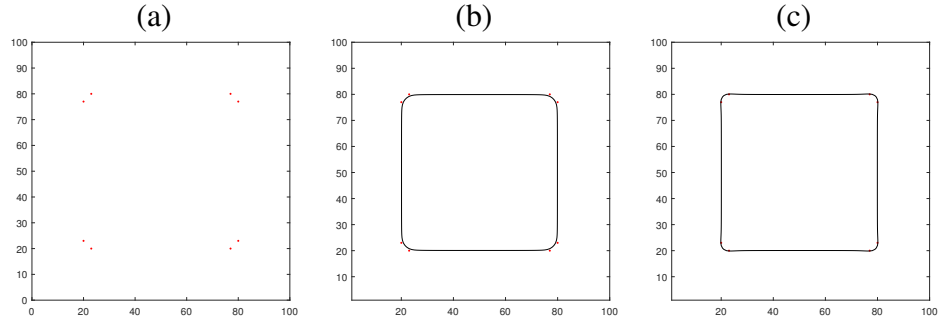


Figure 4.18: By OSM, extremely sparse data: (a) Given data. (b) The recovered result with $\eta = 1$ and (c) with $\eta = 1.5$. Even with extremely sparse data, curvature constraint model can reconstruct the square corners well.

smooth corners. These are due to the fact that the curvature along a straight edge is zero, and smooth corners have smaller curvature than sharp corners for $s = 2$ in the discrete setting.

The next experiment is for the noisy boomerang data, where Gaussian noise with standard deviation 1 is added to the locations of the point cloud. The results with $\eta = 0, 1, 2$ are shown in Figure 4.19. As η gets larger, the two lower corners get recovered better. Even with noisy data, OSM shows a strong competence of recovering the sharp corners.

4.4.4 Three Dimensional Examples

We conclude this section with experiments of reconstruction of surfaces in three dimensional space. We use OSM with $s = 2$ to reconstruct the pyramid, the yoyo, and the ice

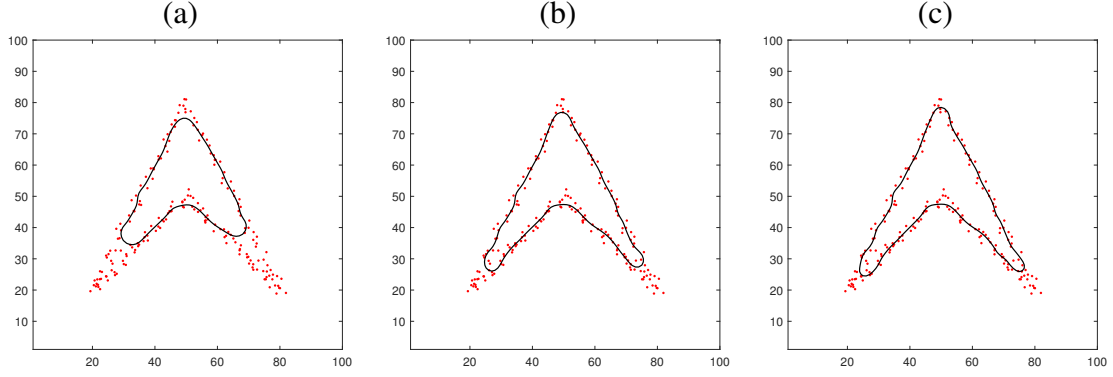


Figure 4.19: By OSM, reconstruction with noisy data: (a) $\eta = 0$, (b) $\eta = 1$, (c) $\eta = 2$. The noise is Gaussian with standard deviation 1. As η gets larger, the two lower corners are better recovered.

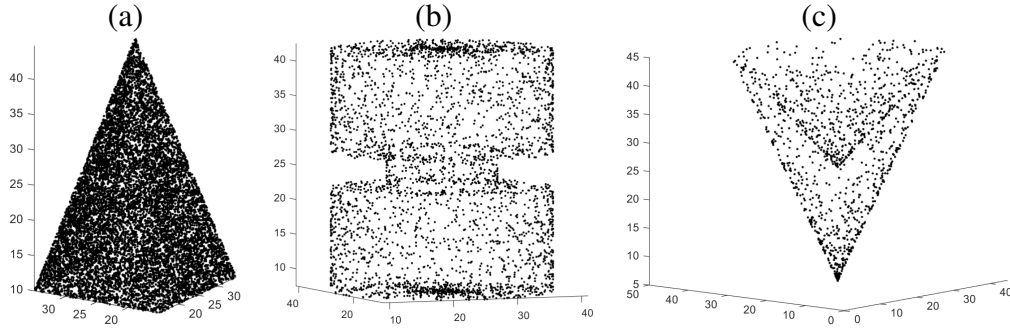


Figure 4.20: Examples of three dimensional point cloud data. (a) A pyramid. (b) A yoyo. (c) An ice cream cone.

cream cone, whose point clouds are shown in Figure 4.20. The data in these examples are concentrated within a cube $[0, 50]^3$. The pyramid has a relatively simple geometry structure: it is convex and its surface only consists of five plans. We can use a large time step $\Delta t = 500$. For the yoyo and the ice-cream cone we use $\Delta t = 100$, since the underlying surfaces have more details, e.g., the neck of the yoyo and the upper concave part of the ice cream cone.

For the pyramid, the reconstructed surfaces with $\eta = 0, 5, 10$ and the comparison of cross sections along $y = 25$ (a middle section) are shown in Figure 4.21. In this case, we see limited improvements of capturing the vertices when the curvature constraint is included.

For the yoyo, the reconstructed surface with $\eta = 0, 5$ and the comparison of cross

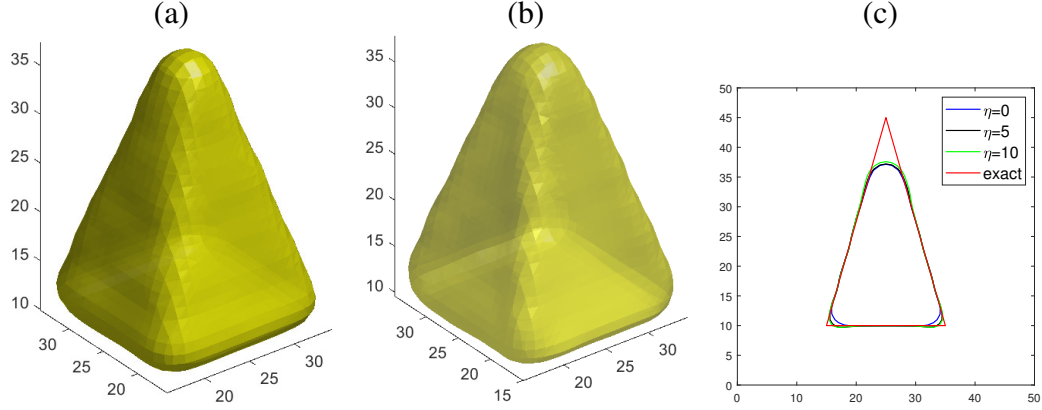


Figure 4.21: Reconstruction of the pyramid by OSM with $s = 2$: (a) Result with $\eta = 0$. (b) Result with $\eta = 10$. (c) Comparison of cross section along $y = 25$.

sections along $y = 25$ are shown in Figure 4.22. The advantage of the curvature term is obvious. With $\eta = 0$, the solution attains the energy balance between surface area and distance to the data at some location away from the middle neck part. Since the curvature at that part is non-zero, given a positive η , the surface further evolves to capture the neck. For comparison, the reconstructed surface by the algorithm in [274] is shown in Figure 4.22 (a). Similarly to the result by OSM with $\eta = 0$, [274] fails to capture the neck part.

The ice cream cone surface consists of two layers and its cross section looks like a boomerang. For this example, if we use $\eta = 0$, the solution shrinks to a point and then disappears. The reconstructed surfaces with $\eta = 5, 10$ and the comparison of cross sections along $y = 25$ are shown in Figure 4.23 (b)-(d). The effect of the value of η on this ice cream cone is similar to that on the boomerang. Results with larger values of η capture better the features of the underlying surface such as corners. The reconstructed surface by the algorithm in [274] is shown in Figure 4.23 (a). [274] recovers the bottom corner better but fails to reconstruct the upper concave part of the surface.

4.5 Summary

In this chapter, we explored the surface reconstruction models based on point cloud data based on two energy functionals. The first one finds the underlying level-set representation

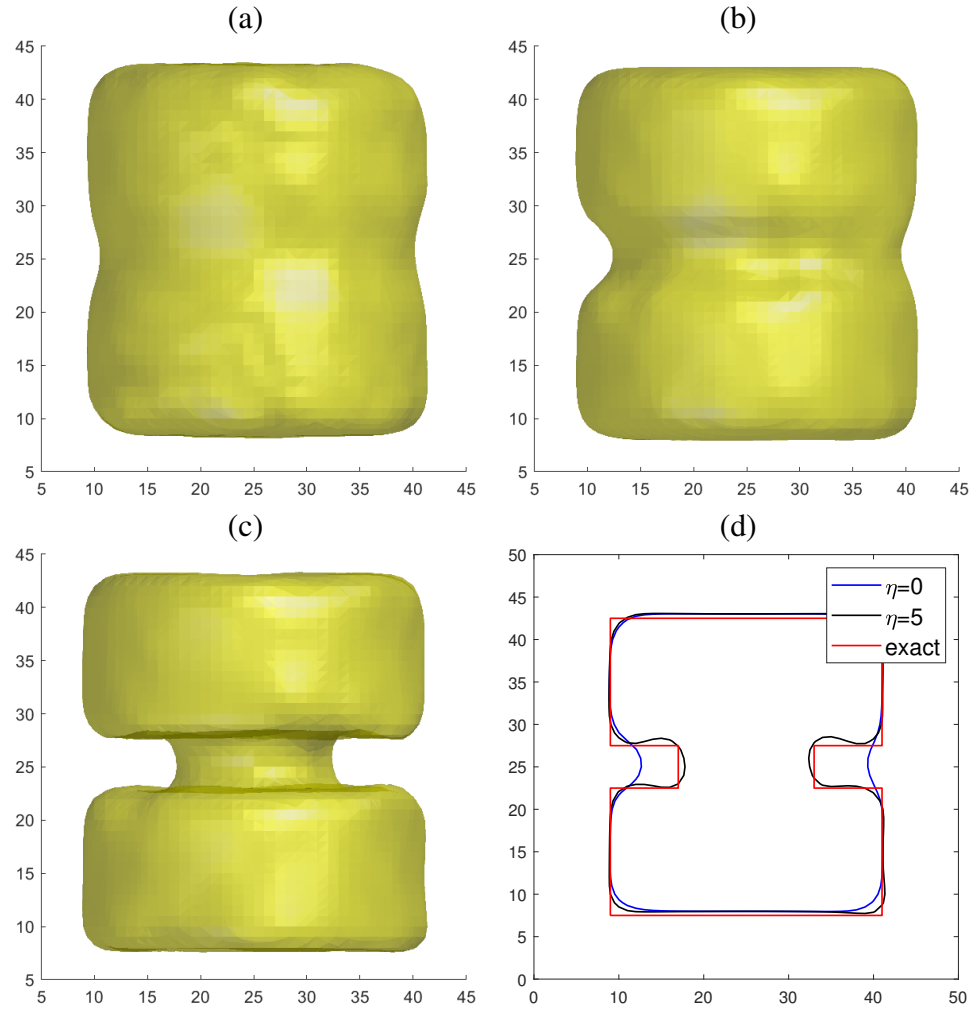


Figure 4.22: Reconstruction of the ice cream cone by the algorithm from [274] and OSM with $s = 2$: (a) Result by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. (b) Result by OSM with $\eta = 0$. (c) Result by OSM with $\eta = 5$. (d) Comparison of cross sections of results by OSM along $y = 25$.

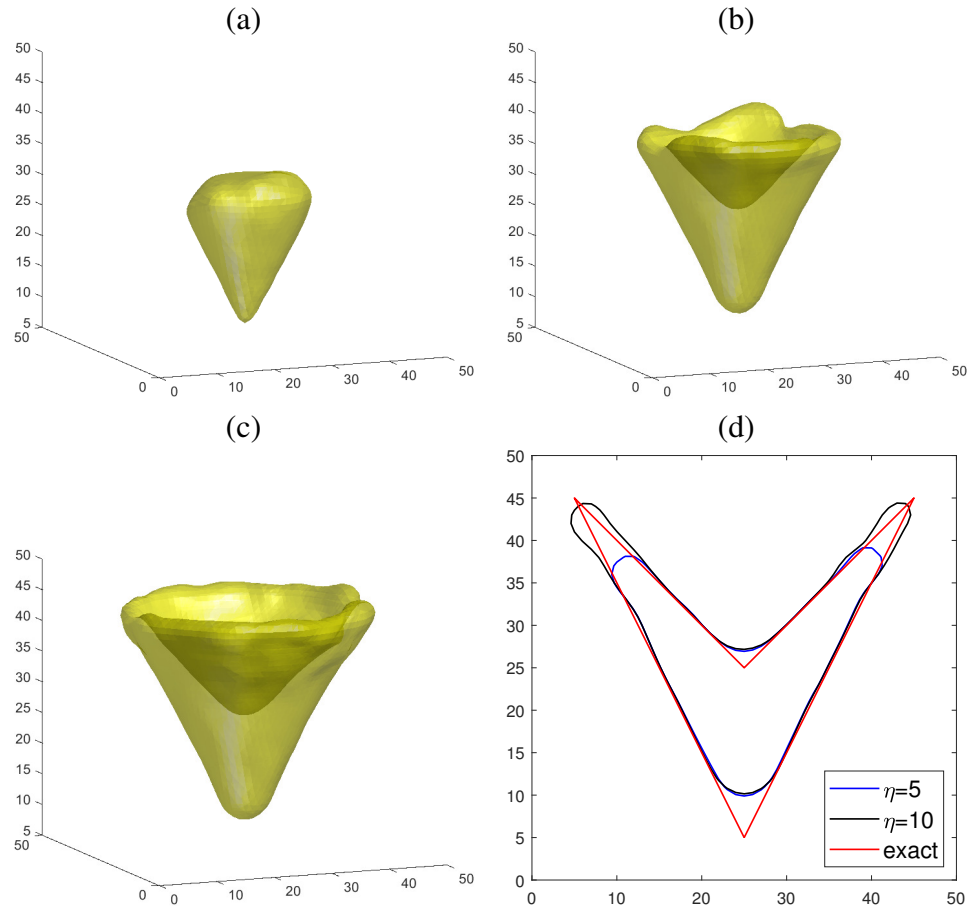


Figure 4.23: Reconstruction of the ice cream cone by the algorithm from [274] and OSM with $s = 2$: (a) Result by the algorithm proposed in [274] with $r_1 = r_2 = 8, r_3 = r_4 = 3$. (b) Result by OSM with $\eta = 5$. (c) Result by OSM with $\eta = 10$. (d) Comparison of cross sections of results by OSM along $y = 25$.

of submanifold by minimizing a distance-weighted surface area. We described two fast algorithms, SIM and ALM, to reconstruct a codimensional 1 submanifold from unstructured point clouds in \mathbb{R}^2 or \mathbb{R}^3 by minimizing the weighted minimum surface energy (Equation 4.2). SIM improves the computational efficiency by relaxing the constraint on the time-step using a semi-implicit scheme. ALM follows an augmented Lagrangian approach and solves the problem by an ADMM-type algorithm. Numerical experiments show that the proposed algorithms are superior at the computational speed, and both of them produce accurate results. Theoretically, we demonstrate the delicate interaction among parameters involved in ALM and show the connections between SIM and ALM. This explains the behaviors of ALM from the perspective of SIM.

The second model combines the distance from surface to the point cloud with a global curvature regularization. Introducing this high-order geometric information allows us to impose geometric features around the corners and to reconstruct concave features of the point cloud better. We find that the interactions between two terms of the functional are subtle. For example, for $s = 1$, since it allows sharp corners, it may be more relaxed around the corners and give shorter length reconstruction. For the curvature term in model (Equation 4.27), larger s gives more weight to the part of the surface which has large curvature, i.e., corners. As a result, the corners of the reconstructed surface are smoothed and extruded out a little bit. For a fast computation, instead of directly solving the complicated terms in its Euler-Lagrange equations, we use a new operator splitting strategy and minimize the energy by a semi-implicit scheme. We also explore an augmented Lagrangian method, which has the advantage of having less parameters compared to other ADMM approaches. Both methods are computationally efficient and produce reliable results in many cases, including those where the point cloud is noisy or sparse. Comparison between OSM and ALM shows advances of OSM in flexibility, stability, and efficiency. Comparing the results of OSM using $s = 1$ and $s = 2$, we find that OSM with $s = 2$ provides better results when reconstructing features of the point clouds. There are a number of extensions to be

considered, including different curvature constraints and using additional information such as surface normal directions to facilitate the reconstruction. Applications to segmentation and image inpainting can also be considered.

CHAPTER 5

COMPLEMENTARY ADAPTATION IN UNDERWATER COLOR CORRECTION

Water absorbs light similarly to an optical filter but with higher variations and complexities [316]. Depending on the dissolved or suspended substances, a liquid medium modifies the spectral power distribution of the transmitted light, such that a strong bluish or greenish color cast dominates the acquired underwater image, e.g., Figure 5.1 (Top). Typically, underwater images have insufficient contrast and unbalanced color distribution [316, 317, 318, 319, 320], hence many image contents, such as patterns and textures are hardly recognizable for human observers. An effective color correction method is needed to recover and enhance these details.

We can understand this task as reversing the process of image formation. The complex factors determining the irradiance on an imaging sensor are often simplified by the Koschmieder model [322]. It expresses the image colors as a convex combination of the unattenuated objects' colors and the veiling light via a scalar transmission map. The veiling light is approximated by various types of dark-channel priors [317, 318, 319], and the estimation of the transmission map is converted to depth computation based on the Beer-Lambert law [323]. Hence, for any combination of a veiling light and a transmission map, the color-corrected image is uniquely determined. These methods are sensitive to the identified veiling lights such that small perturbations on the estimated RGB values of the background trigger visually significant results [319]. More sophisticated models along this direction consider the Jerlov's water types [324, 320] to improve the stability.

Essentially, the goal is to find a color distribution on the image domain that is favorable for a human observer. Different from models of physics, many methods in the literature adjust the image colors based on principles of the human visual system (HVS). One of the important properties of HVS is color constancy: the appearance of the color of an object



Figure 5.1: (Top) Underwater image with heavy green cast. (Bottom) Result of the proposed method. In the middle, several zoomed-in regions are displayed for comparison. The resulted image has enhanced contrast, balanced colors, and many image contents, e.g., the patterns on the swimming shorts, are more recognizable. In this paper, all the underwater images are from the benchmark data set [321].

remains approximately stable under varying illuminations [325]. This chromatic adaptation for instance allows an observer to recognize the brown statue and the blue shorts in Figure 5.1 (Top), even though the image is dominated by a heavy green cast. Theories [326, 327, 328] have been proposed to explain the underlying mechanism from various perspectives including the well-known Retinex theory by Land [53]. It is argued that HVS perceives a scene based on local variation of image lightness rather than an absolute lightness, and this theory induces a huge class of algorithmic interpretations of the adaptation process applied in computer vision, e.g., Multiscale Retinex [329, 330], random-spray Retinex [331], non-local Retinex [332] and many others [54, 55, 333].

In this chapter, we introduce a novel approach for underwater image color correction which converts Figure 5.1 (Top) to (Bottom), whose color distribution is more balanced and compatible with HVS. Instead of the Retinex theory, we present a new mathematical interpretation for the Complementary Adaptation Theory (CAT) first formulated by Gibson [334] in 1937. The key principle is that, the quality of a constantly applied stimulus will be temporarily shifted towards the corresponding complementary quality, thus resulting in a neutral state. This applies not only to HVS, but also to other bilateral sensory processes, e.g., temperature perception. Both Retinex theory and CAT emphasize the importance of relative levels over absolute levels of sensation, yet they are fundamentally different in the following aspects.

- *Mechanism*: The Retinex theory ascribes the color constancy to HVS's ability of estimating the reflectance independent from the illumination, while CAT describes the color constancy as a result from the neutralization of the illumination by a negative sensory process.
- *Role of illuminating color*: In Retinex theory, illumination is treated as unknown and its color can be derived after identifying the reflectance; whereas in CAT, the illuminating light determines the direction and magnitude of the adaptation process.

- *Adapting time*: The Retinex theory was supported by experiments with short-time adapting; in contrast, recent experiments show that the chromatic neutralization predicted by CAT occurs after multiple days [335, 336].

Our method captures these features of CAT and produces a color distribution complying with the long-term chromatic adaptation. Figure 5.2 shows the outline of our method.

The idea is that, we utilize the complementary pairs of the locally approximated color cast to modify the image colors, such that any image colors similar to the color cast are muted, while the others keep their differences relative to the color cast. In other words, we shift the reference color from the chromatic color cast, typically blue and green, to a neutral gray. The resulted color distribution has softer contrast and lower saturation due to the long-term adaptation. Hence, for visualization purposes, we enhance the image while preserving the adapted hues.

In particular, we consider the CIELAB color space and formulate the CAT adaptation process as a Tikhonov-type optimization problem [337]. As a metric space, CIELAB is a subspace of the three dimensional Euclidean space, where the distance between any two colors measures their perceptive difference. Using the CIELAB color difference metric, our optimization model consists of a fidelity term and a regularization characterizing the behavior of CAT adaptation. Then we enhance the adapted color distribution for visualization purposes. We also address some technical issues about CIELAB to improve its uniformity. These modifications are kept minimal so that problematic behaviors are effectively adjusted, and high efficiency is achieved.

5.1 CIELAB Color Space

5.1.1 Basic Notions of CIELAB

Before stating our model, we fix some notations. We denote an arbitrary color by \mathbf{c} , which is specified in the CIELAB color space by its lightness L^* , red-green value: a^* , and yellow-

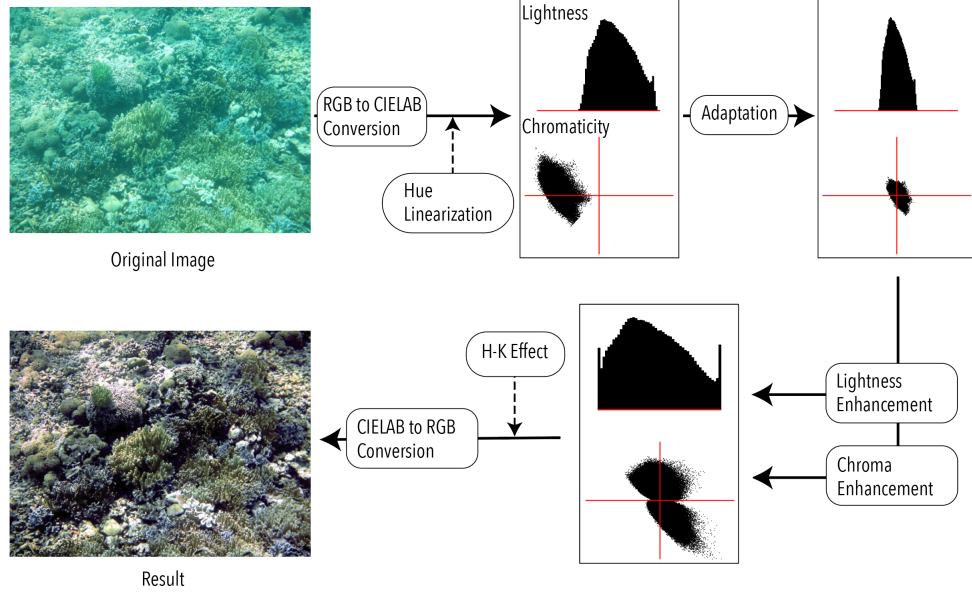


Figure 5.2: Pipeline of the proposed method. The result shown here uses $\eta = 10, \beta = 1/3$ as the model parameters.

blue value: b^* , i.e., $\mathbf{c} = (L^*, a^*, b^*)$. In particular, the range for L^* is $[0, 100]$, where $L^* = 0$ yields black and $L^* = 100$ yields diffuse white. The a^* - b^* section specifies the chromaticity. A positive value of a^* indicates red while a negative value gives green. A positive value of b^* indicates yellow and a negative value gives blue. From CIELAB, one can derive

$$\text{Chroma: } C^* = \sqrt{(a^*)^2 + (b^*)^2}, \quad (5.1)$$

which measures the relative saturation of \mathbf{c} , and

$$\text{Hue angle: } h^\circ = \text{atan2}(b^*, a^*), \quad (5.2)$$

which defines the hue of \mathbf{c} .

Given a pair of colors $\mathbf{c}_i = (L_i^*, a_i^*, b_i^*)$, $i = 1, 2$, the CIELAB color difference between

them is computed by

$$\Delta E^*(\mathbf{c}_1, \mathbf{c}_2) = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}, \quad (5.3)$$

which is simply the Euclidean distance between the CIELAB coordinates of the colors to be compared. This formula suggests that CIELAB color space is designed to be uniform.

The complementary color of \mathbf{c} in CIELAB is denoted by \mathbf{c}^- , which is computed by

$$\mathbf{c}^- = (100 - L^*, -a^*, -b^*). \quad (5.4)$$

On a rectangular image domain $\Omega = [0, W] \times [0, H] \subset \mathbb{R}^2$, $W, H > 0$, we define a color image, or a color distribution as a mapping \mathbf{c} from Ω to the CIELAB color space

$$\mathbf{c}(x, y) = (L^*(x, y), a^*(x, y), b^*(x, y)), \quad (x, y) \in \Omega. \quad (5.5)$$

For a triplet $\sigma = (\sigma_1, \sigma_2, \sigma_3) \in \mathbb{R}^3$ with positive entries, we define the component-wise Gaussian convolution \mathcal{G}_σ applied on a color distribution \mathbf{c} as

$$\begin{aligned} \mathcal{G}_\sigma * \mathbf{c}(x, y) = \\ (\mathcal{G}_{\sigma_1} * L^*(x, y), \mathcal{G}_{\sigma_2} * a^*(x, y), \mathcal{G}_{\sigma_3} * b^*(x, y)), \end{aligned} \quad (5.6)$$

where each component is the ordinary Gaussian convolution with intensity specified by σ_i , $i = 1, 2, 3$. We take mirror reflect for computing the convolved values near the image boundary.

5.1.2 CIELAB Boundary Estimation

The RGB color space is geometrically a cube embedded in the Euclidean space \mathbb{R}^3 , however, the transformation from RGB to CIELAB maps the RGB cube to an irregular shape.

See the illustration in Figure 5.3 (a)–(c). Here, we randomly sample 5×10^5 points in the RGB cube, convert them to the CIELAB space, digitize their lightness coordinates by taking the ceil function, and compute the convex hull of the chromaticity section for each digitized lightness. Hence, the CIELAB gamut is approximated by the union of these convex slices at different lightness levels

$$\bigcup_{L^*=0}^{100} \text{conv}\{V_i(L^*)\}_{i=1}^{N(L^*)}, \quad (5.7)$$

where $V_i(L^*)$ represents a vertex of the convex hull computed using the color samples with digitized lightness L^* , and $\text{conv}\{\cdot\}$ computes the convex hull supported by a finite set of points. This approach offers a simple estimation of the chroma boundary given the lightness L^* and hue angle h° according to basic trigonometry

$$C_{\max}^*(L^*, h^\circ) = \frac{l_{j(h^\circ)} \sin(\rho_{j(h^\circ)})}{\sin(\pi - \theta_{j(h^\circ)} - \rho_{j(h^\circ)})},$$

$$j(h^\circ) \in \{1, 2, \dots, N(L^*)\} \quad (5.8)$$

where $\theta_{j(h^\circ)} < h^\circ < \theta_{j(h^\circ)+1}$, $\theta_{j(h^\circ)}$ is the angle from the positive direction of the a^* -axis to $V_{j(h^\circ)}(L^*)$ in a counter-clockwise orientation, $\theta_{N(L^*)+1}$ takes θ_1 , and $l_{j(h^\circ)}$ is the distance from $V_{j(h^\circ)}(L^*)$ to the origin. Both quantities $l_{j(h^\circ)}$ and $\theta_{j(h^\circ)}$ depend on L^* , and we suppress this notation in Equation 5.8 for simplicity. See Figure 5.3 (d) for an illustration.

5.2 Complementary Adaptation Model in CIELAB

5.2.1 Tikhonov-type Optimization in CIELAB

Given a color distribution $\mathbf{c}_0 = (L_0^*, a_0^*, b_0^*)$ over Ω , we propose the **Complementary Adaptation Model** by defining the adapted color at $(x, y) \in \Omega$ as the minimizer of the following

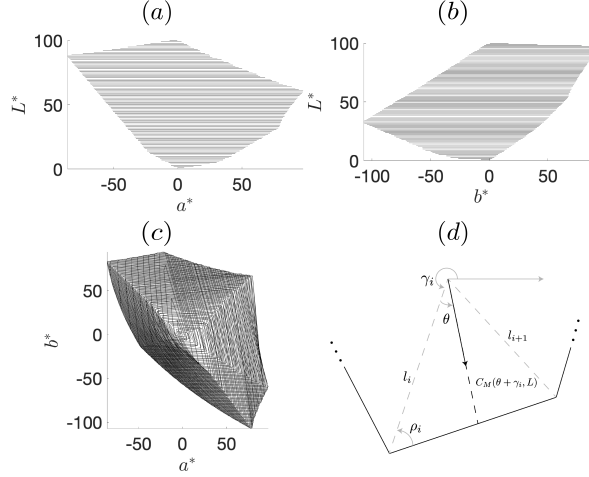


Figure 5.3: (a) CIELAB gamut projection on the L^* - a^* plane. (b) Projection on the L^* - b^* plane. (c) Projection on the a^* - b^* plane. (d) Geometry for computing the chroma upper limit $C_{\max}^*(L, \theta + \gamma_i)$ in the direction of the hue angle $\theta + \gamma_i$ on the lightness level of L . Here the chroma direction falls within the sector $[\gamma_i, \gamma_{i+1}]$, $i = 1, 2, \dots$

optimization problem

$$\begin{aligned} \mathbf{c}_{\text{adapt}}(x, y) = \arg \min_{\mathbf{c} \in \mathbb{R}^3} & (\Delta E^*(\mathbf{c}(x, y), \mathbf{c}_0(x, y)))^2 \\ & + \lambda (\Delta E^*(\mathbf{c}(x, y), (\mathcal{G}_\sigma * \mathbf{c}_0(x, y))^-))^2, \end{aligned} \quad (5.9)$$

where $\lambda > 0$ is a weight parameter, and $\sigma = (\sigma_{L^*}, \sigma_{a^*}, \sigma_{b^*})$ such that $\sigma_{a^*} = \sigma_{b^*}$ and $\sigma_{L^*} = n\sigma_{a^*}$ for some $n > 1$. This is a Tikhonov-type optimization problem consisting of two terms. The first term measures the difference between the given color distribution \mathbf{c}_0 and the adapted color $\mathbf{c}_{\text{adapt}}$, thus it imposes the fidelity condition. The second term models the effect of CAT adaptation process, which acts as a regularization. The proposed color distribution $\mathbf{c}_{\text{adapt}}$ is a balance between the original image and the complementary of the estimated color cast $\mathcal{G}_\sigma * \mathcal{C}$. In this paper, we fix $\sigma_{a^*} = \sigma_{b^*} = \sigma_0 := 0.25(\max(W, H)/2 - 1)$ so that the size of the filter is roughly $\max(W, H)/2$, $n = 3$, and $\lambda = 1$.

Thanks to the simple formula (Equation 5.3) for computing the color difference, (Equation 5.9)

has a unique global minimizer obtained by calculus:

$$\mathbf{c}_{\text{adapt}}(x, y) = (L_{\text{adapt}}^*(x, y), a_{\text{adapt}}^*(x, y), b_{\text{adapt}}^*(x, y)) , \quad (5.10)$$

where

$$\begin{cases} L_{\text{adapt}}^*(x, y) = (L_0^*(x, y) + (100 - \mathcal{G}_{3\sigma_0} * L_0^*(x, y))) / 2 \\ a_{\text{adapt}}^*(x, y) = (a_0^*(x, y) - \mathcal{G}_{\sigma_0} * a_0^*(x, y)) / 2 \\ b_{\text{adapt}}^*(x, y) = (b_0^*(x, y) - \mathcal{G}_{\sigma_0} * b_0^*(x, y)) / 2 \end{cases} \quad (5.11)$$

This formulation shows that the adapted color $\mathbf{c}_{\text{adapt}}(x, y)$ is the midpoint of the image color $\mathbf{c}_0(x, y)$ and the complementary pair of the estimated color cast at (x, y) in the CIELAB space.

Some remarks are needed for the proposed model:

1. *Locality principle*: The adaptation is spatially dependent [338]. Notice that in the second term of the model, the complementary operator is applied to the Gaussian filtered color distribution instead of the original \mathbf{c}_0 . The locality of the adaptation is adjusted by the parameter σ_0 . A greater value of σ_0 implies a larger field of adaptation and the estimated color cast is spatially more uniform. In contrast, a smaller value of σ_0 induces a more focused adaptation and the estimated color cast is more variant. Consequently, the neutralization effect is stronger when σ_0 is small; when $\sigma_0 \rightarrow 0$, the adapted color distribution becomes uniformly neutral gray.
2. *CIELAB gamut consideration*: In practice, the 3-tuple $\mathbf{c}_{\text{adapt}}$ computed by Equation 5.11 stay inside the CIELAB gamut. There are two reasons to support this statement. First, the adapted lightness L_{adapt}^* concentrates around 50, where the chromaticity section of the CIELAB gamut has the most extended domain (Figure Figure 5.3 (a) and (b)). Second, the dominating color casts in underwater images are

mostly blue or green. Observe that the CIELAB gamut (Figure 5.3 (c)) corresponding to the green-blue region only has limited expansion, hence both $\mathcal{G}_{\sigma_0} * a_0^*$ and $\mathcal{G}_{\sigma_0} * b_0^*$ are relatively small. Consequently, the triangle spanned by (a_0^*, b_0^*) and $(\mathcal{G}_{\sigma_0} * a_0^*, \mathcal{G}_{\sigma_0} * b_0^*)$ is most likely contained in the chromaticity domain at the lightness L_{adapt}^* . For robustness, in case $\mathbf{c}_{\text{adapt}}(x, y)$ for some (x, y) falls outside the CIELAB gamut, we keep its adapted lightness and hue angle while shrinking its chroma to the corresponding maximal chroma. Other possible solutions can be found in [339] and [340].

3. *Long-term adaptation:* The proposed color distribution $\mathbf{c}_{\text{adapt}}$ is based on the neutralization of dominant colors, which is a long-term chromatic adaptation that can take multiple days [335, 336]. This is different from the daily experience where the time of adaptation ranges from seconds to a few minutes [341]. Similarly to the experimental setting [335], the Gaussian filtered color distribution can be considered as colored lenses, and the long-term adaptation behavior is modeled by the regularization term. Hence, our model predicts the perceived colors when the observer wears the lenses for a long time and the dominant colors are neutralized by their complementary pairs, respectively.
4. *Connection to other works:* The proposed model (Equation 5.9) also provides a variational substitute for the well-known Gray World (GW) assumption [342], which is a key component in many methods in the literature, e.g., ACE [71]. In [71], Bertalmío et al. connect ACE to the Wilson-Cowan equations [343] from computational neuroscience, where the GW assumption is used to set an absolute neutral state such that only deviations from this level are considered meaningful. Noticing the drawback of using an absolute level, in [344], Bertalmío proposes to replace it with a local average; however, by doing so, no color correction is in action. Our model provides an elegant solution which maintains an effective color correction while avoiding an

absolute reference.

5.2.2 Robust Hue-preserving Image Enhancement

The adapted color distribution $\mathbf{c}_{\text{adapt}}$ represents a long-term result rarely achieved in common life experience. For visualization purpose, we enhance the lightness L_{adapt}^* and the chroma C_{adapt}^* (Equation 5.1) computed using $a_{\text{adapt}}^*, b_{\text{adapt}}^*$, while preserving the adapted hue h_{adapt}° (Equation 5.2) where the dominant color cast has been neutralized.

We enhance the adapted lightness by a linear stretch. The enhanced lightness is denoted by $\widehat{L}_{\text{adapt}}^*$. To keep the transform consistent with the CIELAB gamut, we rescale the chroma of the adapted colors by

$$C_1^*(x, y) = \left(\frac{C_{\text{adapt}}^*(x, y)}{C_{\text{max}}^*(L_{\text{adapt}}^*(x, y), h_{\text{adapt}}^\circ(x, y))} \right) \times C_{\text{max}}^*(\widehat{L}_{\text{adapt}}^*(x, y), h_{\text{adapt}}^\circ(x, y)) . \quad (5.12)$$

which preserves the percentage of the relative saturation of $\mathbf{c}_{\text{adapt}}$. Here $C_{\text{max}}^*(L^*, h^\circ)$ denotes the maximal chroma in the CIELAB gamut when the lightness is L^* and the hue angle is h° , whose computation is detailed in subsection 5.1.2. Notice that the adapted hue angles are unchanged during this rescaling.

For the fixed lightness $\widehat{L}_{\text{adapt}}^*$, we enhance the chroma of the newly obtained color distribution $(\widehat{L}_{\text{adapt}}^*, a_1^*, b_1^*)$ by the following transformation

$$C_2^*(x, y) = \left(\frac{C_1^*(x, y)}{C_{\text{max}}^*(\widehat{L}_{\text{adapt}}^*(x, y), h_{\text{adapt}}^\circ(x, y))} \right)^{1/\eta} \times C_{\text{max}}^*(\widehat{L}_{\text{adapt}}^*(x, y), h_{\text{adapt}}^\circ(x, y)) . \quad (5.13)$$

We note that this is a gamma correction applied to the percentage of relative saturation. See Figure 5.4 (a). Here, $\eta \geq 1$ is the enhancing parameter. When η increases, a stronger enhancement is applied, and when $\eta = 1$, the gamma function reduces to the identity map.

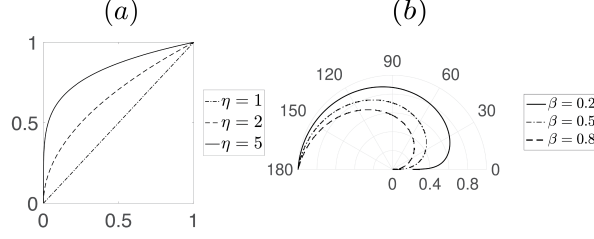


Figure 5.4: (a) Gamma function used for chroma enhancement (Equation 5.13) with varying values of η . (b) Robust factor (Equation 5.14) for suppressing the noisy hues with varying values of β .

To improve stability, for some $0 < \beta \leq 1$, we define

$$\text{Robust factor: } F(\theta) = \left(\frac{\theta}{180^\circ} \right)^\beta, \quad (5.14)$$

for any $\theta \in [0, 180^\circ]$,

whose behavior is shown in Figure 5.4 (b). We propose the robust hue-preserving enhancement of the adapted color distribution $\mathbf{c}_{\text{adapt}}$ by

$$\widehat{\mathbf{c}}_{\text{adapt}}(x, y) = (\widehat{L}^*_{\text{adapt}}(x, y), \widehat{a}^*_{\text{adapt}}(x, y), \widehat{b}^*_{\text{adapt}}(x, y)), \quad (5.15)$$

where

$$\begin{cases} \widehat{a}^*_{\text{adapt}}(x, y) = F(\theta(x, y))a_2^*(x, y) \\ \widehat{b}^*_{\text{adapt}}(x, y) = F(\theta(x, y))b_2^*(x, y) \end{cases}. \quad (5.16)$$

and $\theta(x, y)$ denotes the hue angle difference between the image color and the estimated color cast at (x, y) . Notice that by multiplying the robust factor, when $\theta(x, y) \approx 0^\circ$, i.e., the hue angle difference between the image color and the estimated color cast is small, $\widehat{\mathbf{c}}_{\text{adapt}}(x, y)$ becomes almost achromatic. As for image colors deviating from the estimated color cast at the same locations, the differences are emphasized.

5.2.3 Improvement on the Uniformity of CIELAB

The main purpose of the CIELAB as an alternative to RGB is to quantify the perceptive color difference. This is only approximate due to the intrinsic complexity and non-linearity of the HVS. In this work, we employ two simple modifications to achieve a better uniformity.

Adjustment in the Blue Region

As known to many researchers [345, 346, 347, 348], the blue region of CIELAB, which roughly corresponds to the subset of colors with hue angles ranging from 250° to 300° , is not hue-linear. It means that, with the CIELAB lightness and hue angle fixed, increasing the CIELAB chroma yields a perceivable hue-shift.

In this work, we propose to address this technical problem by applying the following transform before solving for the adapted color distribution via Equation 5.9:

$$h_{\text{adjust}}^\circ = h^\circ - \mu^\circ \times \sqrt{\frac{(C^*)^m}{(C^*)^m + 10^m}} \times \exp\left(-\left(\frac{h^\circ - 275^\circ}{25^\circ}\right)^2\right). \quad (5.17)$$

This formula is modified from [349], which adjusts the hue angle by a product of three factors. The first factor μ° denotes the maximal distorted hue angle. The second factor predicts the increase of the hue rotation from neutral, i.e., $C^* = 0$ until around $C^* = 10$ and remains constant in the high chroma region [349]. The last factor restricts the hue adjustment within the region $275^\circ < h^\circ < 300^\circ$. We fix $\mu^\circ = 45^\circ$ and choose $m = 7$ in this paper. For a more precise hue adjustment based on a look-up-table, we refer the readers to [347].

For underwater images, the pre-processing (Equation 5.17) is especially important, since the general color distributions concentrate around the blue region. In Figure 5.5, we

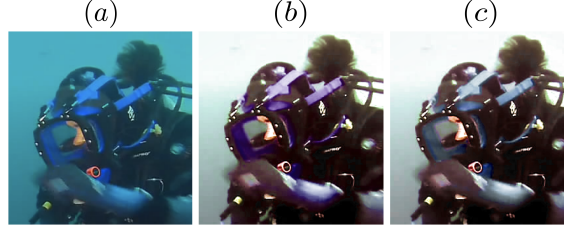


Figure 5.5: Pre-processing by hue angle adjustment in the blue region of CIELAB. (a) Part of an underwater image. (b) Proposed method without the pre-processing (Equation 5.17). (c) Proposed method with the pre-processing. With the adjustment, the blueness on the strap is preserved.

apply the proposed method to an underwater image (a) without the hue adjustment (Equation 5.17) and the blue goggle straps turn into purple (b). With the hue correction (c), we observe that the blueness is correctly preserved. Hence, including the adjustment (Equation 5.17) as a pre-processing compensates for the distortion of the hue angle as we enhance the chroma.

The Helmholtz-Kohlrausch Effect

The Helmholtz-Kohlrausch (H-K) effect is a perceptual phenomenon where the perceived lightness of a color with increasing saturation is brighter [350]. To enhance the chroma (Equation 5.16) while keeping the perceived lightness unchanged, we need to adjust $\widehat{L}^*_{\text{adapt}}$. For an arbitrary color $\mathbf{c} = (L^*, a^*, b^*)$ in CIELAB, the perceived lightness $L^*_{\text{H-K}}$ when considering the H-K effect can be estimated by [351]

$$L^*_{\text{H-K}} = L^* + (2.5 - 0.025L^*)g(h^\circ)C^*, \quad (5.18)$$

where

$$g(h^\circ) = 0.116 \times \left| \sin \left(\frac{h^\circ - 90^\circ}{2} \right) \right| + 0.085. \quad (5.19)$$

$$(5.20)$$

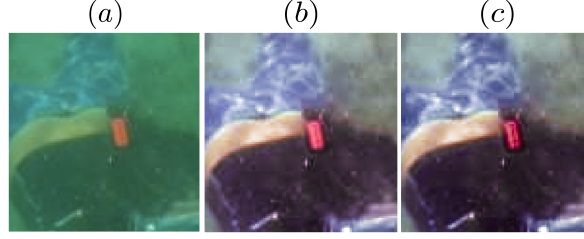


Figure 5.6: Post-processing considering the HK-effect. (a) Zoom-in of part of the underwater image in Figure 5.1. (b) Proposed method without the post-processing (Equation 5.18). (c) Proposed method with the post-processing. Post-processing considering the HK-effect reduces over-exposure.

Hence, assuming that $\widehat{L}_{\text{adapt}}^*$ corresponds to the perceived lightness before the chroma enhancement, the associated CIELAB lightness after the chroma enhancement is computed by inverting (Equation 5.18), which gives

$$(\widehat{L}_{\text{adapt}}^*)_{\text{adjust}} = \frac{\widehat{L}_{\text{adapt}}^* - 2.5g(\widehat{h}_{\text{adapt}}^\circ)\widehat{C}_{\text{adapt}}^*}{1 - 0.025g(\widehat{h}_{\text{adapt}}^\circ)\widehat{C}_{\text{adapt}}^*}. \quad (5.21)$$

Similar improvement is also considered in [340] for food image enhancement.

The post-processing in regard to the HK-effect (Equation 5.18) addresses the over-exposure caused by the enhancing saturation. In Figure 5.6, we focus on a zoomed-in region of an underwater image (a) and show the result without the post-processing (b) as well as the processed one (c). Comparing these results, we observe that when the HK-effect is considered, the image contrast is also improved. See the patterns on the shorts and the red tube.

5.3 Numerical Experiments

5.3.1 General Examples

The proposed method is flexible and adaptive to different image contents. It yields improvements on the image contrasts and color balance which are typically degraded in underwater images. In Figure 5.7, we demonstrate various examples where underwater images are

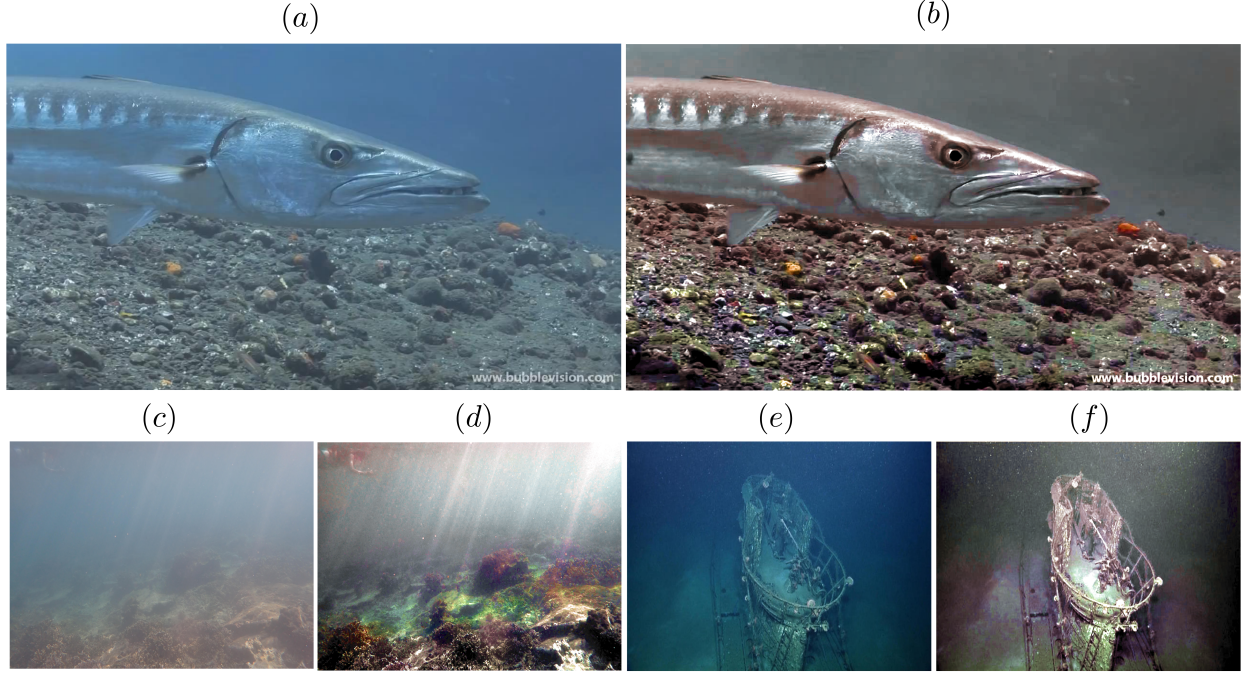


Figure 5.7: General examples of the proposed method. (a) A typical underwater image showing dominating blue cast, and the contrast is relatively low. (b) Result of the proposed method applied to (a) removes the blue cast and enhances the textures on the riverbed. (c) A blurry underwater image where objects are hardly visible. (d) Result of the proposed method applied to (c) which shows vibrant colors and sharp objects' boundaries. (e) A deep underwater image commonly seen in field exploration. (f) Result of the proposed method applied to (e) which renders the details of the structure of interest.

used for (a) submarine biology, (c) recreational purposes, and (e) field exploration. Given possible differences in the imaging environment and devices, our method shows consistent and stable behaviors in terms of removing the color casts and enhancing the image quality, and the corresponding processed results are in (b), (d), and (f).

5.3.2 Different Underwater Color Cast

Underwater imaging environment is complicated and various conditions can affect the chromatic attributes of the color cast. In Figure 5.8, we show the stability of our proposed method for underwater images with different color casts. The scene in the first row shows strong blue veiling light (hue angles concentrating around 210°), the one in the second row has a yellow color cast (around 95°), and the third is dominated by a green color

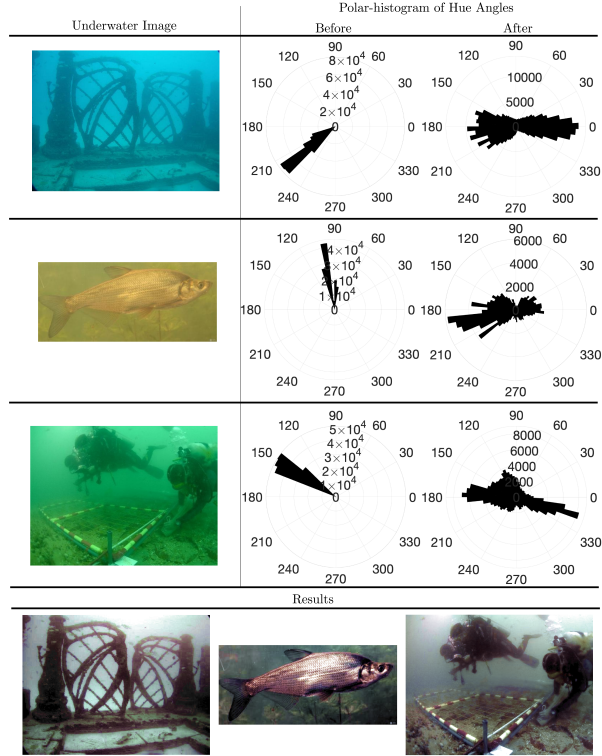


Figure 5.8: The proposed method shows consistent performance for underwater images with different color casts. For each underwater image in the first 3 rows, we show their hue angle distributions in the second column; in the third columns we show the distribution of the final results, which are displayed in the last row. In all cases, we have $\eta = 6$ and $\beta = 1/4$.

(around 150°). In the third column, we observe that the hue angles of the processed results are more spread out, and the resulted images are displayed in the last row. Although the color casts in the original images are distinct, the colors different from the estimated color casts are preserved and emphasized in the final results.

5.3.3 Necessity of the Robust Factor

The procedure (Equation 5.13) enhances the residual colors after the dominating cast is neutralized. Specifically, the absolute change of the hue angle can be expressed as

$$|h_{\text{adapt}}^{\circ}(x, y) - h_0^{\circ}(x, y)| = \frac{180^{\circ}}{\pi} \arccos \left(\frac{a_{\text{adapt}}^*(x, y)a_0^*(x, y) + b_{\text{adapt}}^*(x, y)b_0^*(x, y)}{C_{\text{adapt}}^*(x, y)C_0^*(x, y)} \right). \quad (5.22)$$

Let $C_{\mathcal{G}}^*(x, y) = \sqrt{(\mathcal{G}_{\sigma_0} * a_0^*(x, y))^2 + (\mathcal{G}_{\sigma_0} * b_0^*(x, y))^2}$ be the chroma of the estimated color cast at (x, y) , $\rho(x, y) = C_0^*(x, y)/C_{\mathcal{G}}(x, y)$ as the ratio of image chroma and color cast chroma, and $\gamma(x, y) = (a_0^*(x, y)(\mathcal{G}_{\sigma} * a_0^*)(x, y) + b_0^*(x, y)(\mathcal{G}_{\sigma} * b_0^*)(x, y)) / (C_0^*(x, y)C_{\mathcal{G}}^*(x, y))$ as a measure of the hue angle difference between the image color and the color cast, then Equation 5.22 can be rewritten as

$$\frac{180^{\circ}}{\pi} \arccos \left(\frac{\rho(x, y) - \gamma(x, y)}{\sqrt{\rho^2(x, y) - 2\gamma(x, y)\rho(x, y) + 1}} \right). \quad (5.23)$$

Notice that when the image color and the estimated color cast have similar hue angles, i.e., $\gamma(x, y) \approx 1$, Equation 5.23 shows that

$$|h_{\text{adapt}}^{\circ}(x, y) - h_0^{\circ}(x, y)| \approx \begin{cases} 180^{\circ}, & \text{if } \rho(x, y) < 1 \\ 90^{\circ}, & \text{if } \rho(x, y) = 1 \\ 0^{\circ}, & \text{if } \rho(x, y) > 1 \end{cases}, \quad (5.24)$$

which is independent of the lightness. This implies that a direct enhancement as in Equation 5.13 is very sensitive to the ratio of the image chroma and color cast chroma.

Such instability will cause chromatic noise in areas where the colors are slightly different from the estimated color cast, and typically this happens when the underwater image contains a large portion of background, e.g., Figure 5.9 (a). In Figure 5.9 (b), we show

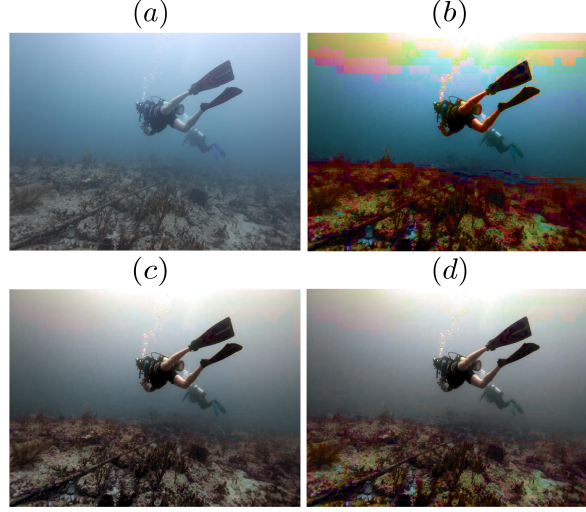


Figure 5.9: (a) Original underwater image. (b) Proposed method without applying the robust factor (Equation 5.14) ($\eta = 8$). (c) Proposed method without applying the robust factor ($\eta = 2$). (d) Proposed method with the robust factor ($\eta = 8, \beta = 1/3$). Using the robust factor suppresses the background noisy colors while enhancing the saturation of the other regions.

the image resulted from the direct enhancing (Equation 5.13) using $\eta = 8$, which presents noisy colors in the background region. A possible remedy is to use smaller values of η , however, this will also subdue the saturation of regions which deserve enhancing. For example, in Figure 5.9 (c) where we use $\eta = 2$, although the noisy colors in the background are suppressed, the riverbed becomes almost achromatic. By using the proposed robust factor (Equation 5.14), the corrected image as shown in Figure 5.9 (d) reduces the noise while maintaining a more saturated rendering outside the background region.

5.3.4 Behaviors of the Saturation Parameter η

The chroma enhancement parameter η (Equation 5.13) allows flexible adjustment of the image saturation. In Figure 5.10, fixing $\beta = 1/3$, we apply the proposed method to the underwater image (a) using $\eta = 2$, $\eta = 4$ and $\eta = 10$, which are shown in (b), (d), and (d), respectively. When we increase the parameter η , the saturated green color cast in the original image keeps muted, as it is neutralized before the chroma enhancement. As for the objects of colors different from the color cast, e.g., the string and statue, they become

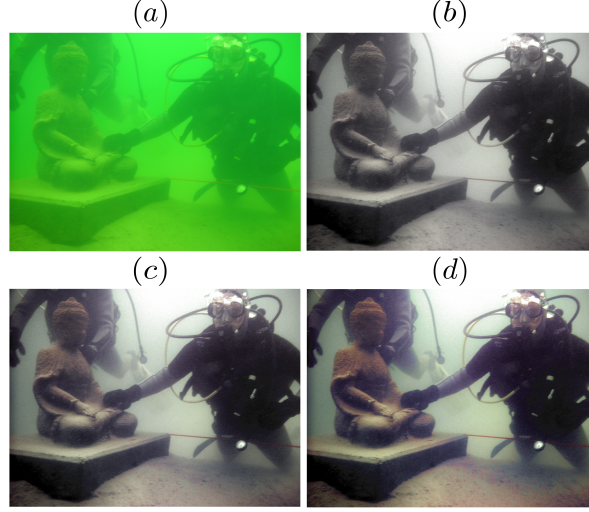


Figure 5.10: Effect of the chroma enhancement parameter η . (a) Original underwater image. (b) Result with $\eta = 2$. (c) $\eta = 4$. (d) $\eta = 10$. Here we fix $\beta = 1/4$.

more recognizable when greater values of η are applied. This example demonstrates two features of our method.

First, objects of smaller scales compared to the radius of the Gaussian kernel used in Equation 5.9 are the most distinguishable in the results. This is due to the fact that, the pixels of these objects have little impact on the estimated color cast, thus the complimentary pairs of the objects' colors will not contribute to the neutralization process according to CAT. Consequently, they preserve most of their chromatic properties and get emphasized after the neutralization of the color cast and the chroma enhancement. For instance, see the red string, the white oxymeter, and the texture of the sand.

Second, although enhancing the saturation by Equation 5.13 is global, the background color remain relatively muted compared to others during this process. When we increase the value of η , the red string and brown statue become more saturated than the green background. This can render the objects in the scene more distinguishable and help improve the image contrast in a global scale.

5.3.5 Qualitative Comparison

We compare our proposed method to some of the state-of-art approaches in the literature. They are designed either specifically for underwater images, or for color constancy for general color images. On the original image in Figure 5.11 (a), we compare Zhao et al. [352] (shown in (b)), Peng et al. [319] (shown in (c)), Histogram Equalization (shown in (d)), Limare et al. [353] (shown in (e)), Automatic Color Correction (ACE) [71, 354], Local Color Correction [355] (shown in (g)), Multiscale Retinex [329, 330] (shown in (h)), and the proposed method in (i). We see that these methods exhibit different chromatic properties in their results.

Both (b) and (c) are obtained from underwater-image-specific approaches based on the Koschmieder model. They differ from each other by the techniques used for background light and transmission map estimation. As pointed out in [319], these estimated quantities determine the results, and in many cases, such relation is very sensitive. With careful combination of different priors and estimations, the result in (c) shows better color restoration on some region of the statue and the riverbed compared to (b).

The methods used in the second row manipulate the image histogram. For (d), we see the typical over-saturation in HE. The method in (e), which aims at enhancing the dynamical ranges of the RGB-channels, does not show effective color balancing in this example. As a localized version of HE, (f) renders more realistic colors compared to (d).

The method for (g) is based on a nonlinear filter applied in the HSL color space, which demonstrate enhancement on the brightness and saturation, yet the green cast is not removed. The Multiscale Retinex used in (h) improves the image brightness and makes many textures visible, but the colors are still biased toward green.

The proposed method in (i) shows distinct visual perception from the others. First, the strong green cast in the original underwater image is effectively removed. This renders a neutral background and recovers realistic tones for the statue, which is perceived as white in (a). Second, the colors for small-scale textures become apparent. For example, we clearly

see the brownish mud around the statue and the color variations on the riverbed. Third, because of the neutralization of the background and the enhancement on the small scale contents, our result shows better contrast improvement. Notice the head and shoulder of the statue, as well as the patterns on the pottery.

5.3.6 Quantitative Evaluation and Comparison

In this set of experiments, we evaluate the performances of the proposed algorithm and other methods in the literature using two measures: Underwater Color Image Quality Evaluation metric (UCIQE) [317] and Underwater Image Quality Measure (UIQM) [356]. These metrics are specifically designed to evaluate the quality of underwater images. Both UCIQE and UIQM are linear combinations of certain image attributes such as colorfulness, saturation, and contrast, whose coefficients are statistically derived. Higher values of these metrics indicate better image qualities.

Figure 5.12 collectively shows the results from Histogram Equalization, Peng et al. [319], Automatic Color Enhancement (ACE) [71], and the proposed method, where the underwater images have various contents and complexities. Our method performs consistently the best measured by UIQM, and the values of UICQE for some of our results are the highest. Among all the methods in comparison, HE produces the most colorful results, yet some of which are overly saturated. The method proposed by Peng et al. performs well when the veiling light color is correctly estimated. ACE is a local HE in principle, hence we observe similar chromatic features between them. Compared to HE, ACE produces more natural color distributions. Among the results from the proposed method, observe that a common characteristic is that the strong color casts in the original underwater images are neutralized. This feature induces a visual effect that the objects against the original saturated background have sharper boundaries.

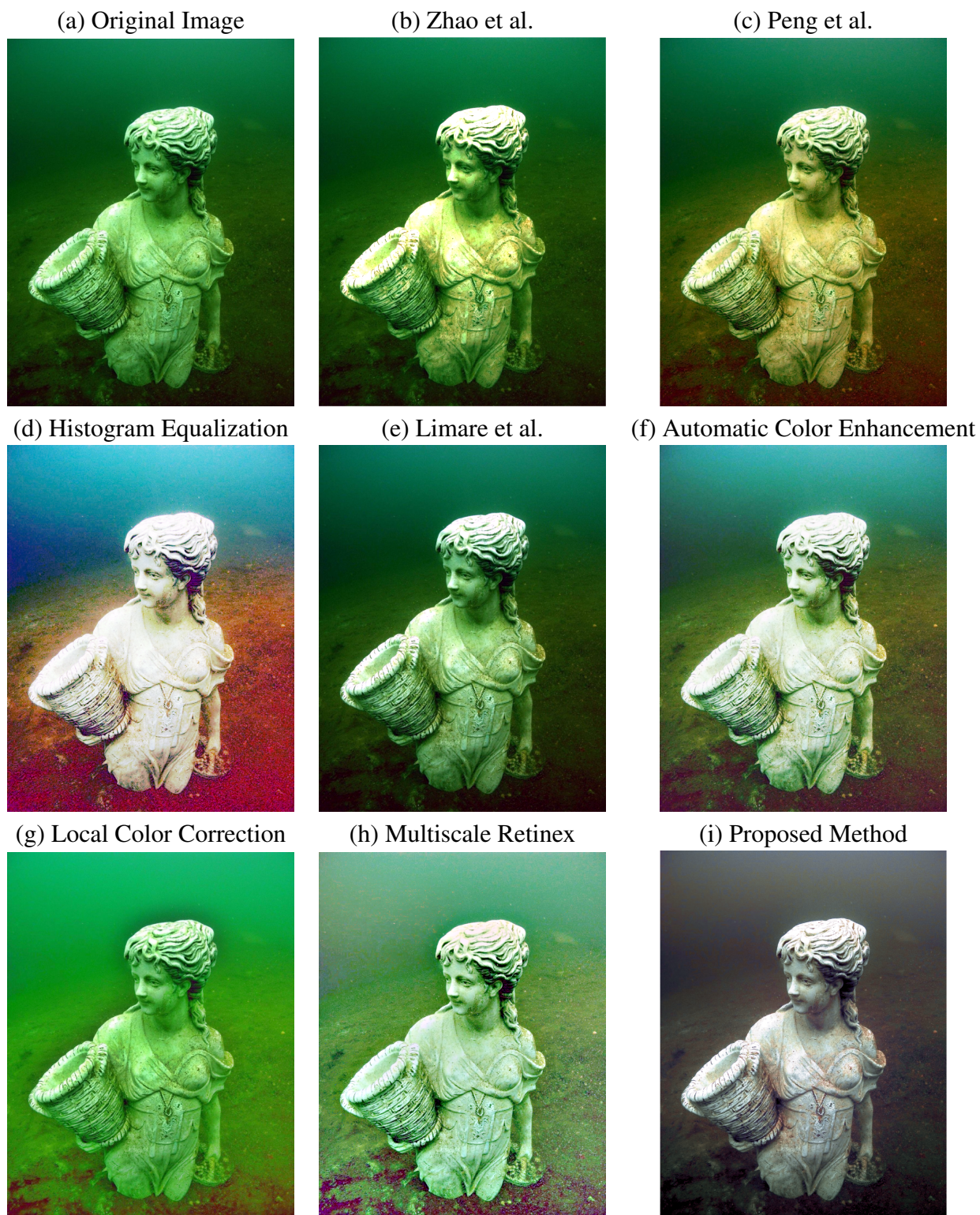


Figure 5.11: Qualitative comparison of different methods (a) Original underwater image. (b) Zhao et al. [352] (c) Peng et al. [319] (d) Histogram Equalization (HE) (e) Limare et al. [353] (f) Automatic Color Enhancement (ACE) [71, 354] (g) Local Color Correction [355] (h) Multiscale Retinex [329, 330] (i) Proposed Method.

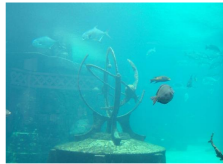
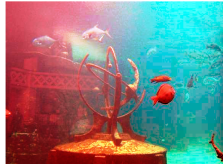
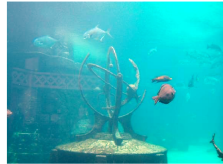
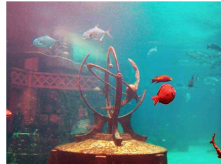
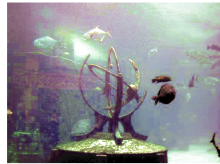





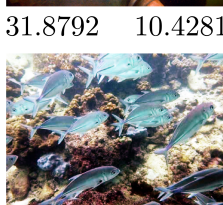
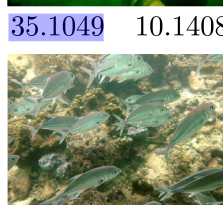
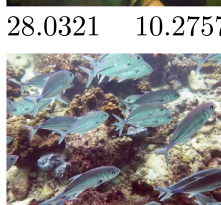
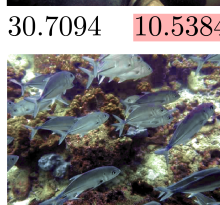
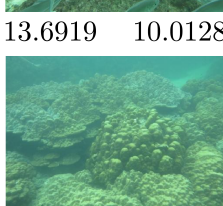
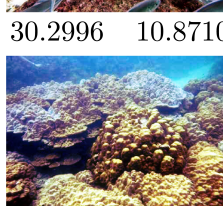
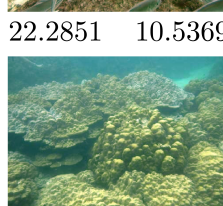
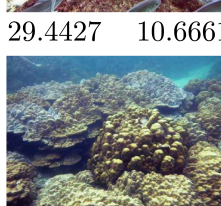


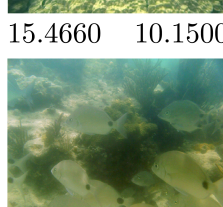

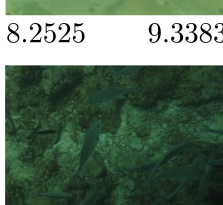

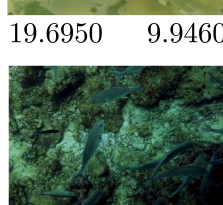
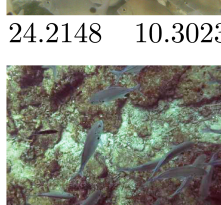
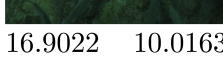



Underwater Image	HE		Peng et al.		ACE		Proposed	
								
15.4562 9.7401	34.7763 10.5910	20.6511 10.1054	28.0375 10.4630	31.2793 10.6749	13.0788 9.4481	31.8792 10.4281	35.1049 10.1408	28.0321 10.2757
								
13.6919 10.0128	30.2996 10.8710	22.2851 10.5369	29.4427 10.6661	31.3955 10.8797				
10.6763 9.6017	32.7145 10.8361	15.4660 10.1500	28.4755 10.5972	33.1036 10.8481				
8.2525 9.3383	31.5581 10.5508	19.6950 9.9460	24.2148 10.3023	29.2597 10.5969				
16.9022 10.0163	30.9155 10.8021	29.0229 10.6899	24.4996 10.6001	31.8314 10.8797				

Figure 5.12: Quantitative evaluation and comparison. We compare our methods with Histogram Equalization (HE), Peng et al. [319], and Automatic Color Enhancement (ACE) [71, 354]. The quality of each image is evaluated by UCIQE [317] (left, blue marks the best) and UIQM [356] (right, red marks the best). In all the cases, we use $\eta = 10$ and $\beta = 1/4$ for the proposed method.

5.4 Conclusion

In this chapter, we presented a new mathematical interpretation of the complimentary adaptation theory proposed by Gibson in 1937. As an alternative of the well-known Retinex theory for understanding the color constancy, CAT emphasizes the neutralization function of the complementary pair of the lasting stimulus rather than HVS's competence of identifying the reflectance. We modeled this adaptation process as a Tikhonov-type optimization problem in the CIELAB color space. This is a simple model which produces the adapted colors as a balance between the original underwater image and the complimentary pair of the estimated color cast. We overcame the lack of uniformity of CIELAB by employing two techniques: a pre-processing compensating the hue-distortion in the blue region, and a post-processing addressing the H-K effect. Numerically, we demonstrated the necessities of the introduced techniques and qualitatively compared our model with some of the state-of-art methods for underwater images. The proposed method shows superior stability when dealing with various underwater environments and recovers realistic colors compatible with the visual perception.

We also notice a more significant data-dependence in this task compared to the previous chapters' materials. The Tikhonov-type optimization model sets up an elegant framework for characterizing the color-balancing behaviors based on the complimentary adaptation theory. However, the visual effects are determined by the CIELAB colorspace, which was developed based on experiments. Like other color spaces, CIELAB has its own limitations, and to address them, additional parameters estimated from experimental data are necessary. This brings up an important issue of model-based representation in problems with less structured data: how to take advantage of the data efficiently to improve the model? To answer this, in the following two chapters, we will discuss data-driven approaches.

CHAPTER 6

AUTOMATIC PDE IDENTIFICATION FROM NOISY DATA

Partial Differential Equations (PDEs) are used to model various real-world phenomena in science and engineering. Numerical solvers for PDEs and analysis of various properties of the solutions have been widely studied in the literature. In this chapter, we focus on the inverse problem: Given a set of time-dependent noisy data, how to identify the governing PDE.

Let the given noisy time-dependent discrete data set be

$$\mathbf{D} := \{U_{\mathbf{i}}^n \in \mathbb{R} \mid n = 0, \dots, N; \mathbf{i} = (i_1, \dots, i_d) \text{ with } i_j = 0, \dots, M-1, j = 1, \dots, d\} \quad (6.1)$$

for sufficiently large integers $N, M \in \mathbb{N}$, where \mathbf{i} is a d -dimensional spatial index of a discretized domain in \mathbb{R}^d , and n represents the time index at time t^n . The objective is to find an evolutionary PDE of the form

$$\partial_t u = f(u, \partial_{\mathbf{x}} u, \partial_{\mathbf{x}}^2 u, \dots, \partial_{\mathbf{x}}^k u, \dots), \quad (6.2)$$

which represents the dynamics of the given data \mathbf{D} . Here t is the time variable, $\mathbf{x} = [x_1, \dots, x_d] \in \mathbb{R}^d$ denotes the space variable, and $\partial_{\mathbf{x}}^k u$ denotes the set of partial derivatives of u with respect to the space variable of order k for $k = 0, 1, \dots$, i.e., $\partial_{\mathbf{x}}^k u := \left\{ \frac{\partial^k u}{\partial x_1^{k_1} \partial x_2^{k_2} \dots \partial x_d^{k_d}} \mid k_1, \dots, k_d \in \mathbb{N}, \sum_{j=1}^d k_j = k \right\}$. We assume that f is a polynomial of its arguments so that the right-hand side of Equation 6.2 is a linear combination of linear and nonlinear differential terms. The model in Equation 6.2 includes a class of parametric PDEs where the parameters are the polynomial coefficients in f .

Parameter identification in differential equations and dynamical systems has been con-

sidered by physicists or applied scientists. Earlier works include [357, 358, 359, 360, 361, 362, 363], and among which, [358, 363] considered the PDE model as in Equation 6.2. Two important papers [364, 365] used symbolic regression to recover the underlying physical systems from experimental data. Recently, sparse regression and L_1 -minimization were introduced to promote sparsity in the identification of PDEs or dynamical systems [366, 367, 368, 369]. In [366], Brunton et al. considered the discovery of nonlinear dynamical systems with sparsity-promoting techniques. The underlying dynamical systems are assumed to be governed by a small number of active terms in a prescribed dictionary, and sparse regression is used to identify these active terms. This sparse regression approach's extensions can be found in [370, 371, 372]. In [367], Schaeffer considered the problem of PDE identification using the spectral method and focused on the benefit of using L_1 -minimization for sparse coefficient recovery. The identification of dynamical systems with highly corrupted and undersampled data are considered in [373, 374]. In [368], Rudy et al. proposed identifying PDEs by solving the L_0 -regularized regression followed by a post-processing step of thresholding. Sparse Bayesian regression was considered in [375] for the recovery of dynamical systems. This series of work focused on the benefit of using L_1 -minimization to resolve dynamical systems or PDEs with specific sparse pattern [376]. Another related problem is to infer the interaction law in a system of agents from the trajectory data. In [377, 378], nonparametric regression was used to predict the interaction function, and a theoretical guarantee was established. Another category of methods uses deep learning [379, 380, 381, 382, 383, 384, 385]. In [369], Identifying Differential Equation with Numerical Time evolution (IDENT) was proposed. It is based on the convergence principle of numerical PDE schemes. LASSO is used to find a candidate set efficiently, and the correct PDE is identified by computing the numerical Time Evolution Error (TEE). Among all the PDEs from the candidate set, the one whose numerical solution best matches the given data dynamics is chosen as the identified PDE. When the given data are contaminated by noise, the authors used a Least-Square Moving Average method to denoise the data as a

pre-processing step. When the coefficients vary in the spatial domain, a Base Element Expansion (BEE) technique was proposed to recover the varying coefficients.

We will first focus on the numerical aspects of the data-driven PDE modeling problem by introducing the work proposed in [386], and then we switch to the theoretical aspects where the convergence of ℓ_1 -norm regularized identification method is concerned.

6.1 Data Organization and Denoising

6.1.1 Data Organization and Notations

Let the time-space domain be $\Omega = [0, T] \times [0, X]^d$ for some $T > 0$ and $X > 0$. Suppose the noisy data \mathbf{D} are given as (Equation 6.1) on a regular grid in Ω , with time index $n = 0, \dots, N$, $N \in \mathbb{N}$ and spatial index $\mathbf{i} \in \mathbb{I}$, where $\mathbb{I} = \{(i_1, \dots, i_d) \mid i_j = 0, \dots, M-1, j = 1, \dots, d, M \in \mathbb{N}\}$. Denote $\Delta t := T/N$ and $\Delta x := X/(M-1)$ as the time and space spacing in the given data, respectively.

At the time t^n and the location $x_{\mathbf{i}}$, the datum is given as

$$U_{\mathbf{i}}^n = u(\mathbf{x}_{\mathbf{i}}, t^n) + \varepsilon_{\mathbf{i}}^n, \quad (6.3)$$

where $t^n := n\Delta t \in [0, T]$, $\mathbf{x}_{\mathbf{i}} := (i_1\Delta x, \dots, i_d\Delta x) \in [0, X]^d$, and $\varepsilon_{\mathbf{i}}^n$ is i.i.d. random Gaussian noise with mean 0. For $n = 0, 1, \dots, N-1$, we vectorize the data in all spatial domains at time t_n , and denote it as $U^n \in \mathbb{R}^{M^d}$. Concatenating the vectors $\{U^n\}_{n=0}^{N-1}$ vertically gives rise to a long vector $U \in \mathbb{R}^{NM^d}$.

The underlying function f in Equation 6.2 is assumed to be a finite order polynomial of its arguments:

$$f(u, \partial_{\mathbf{x}}u, \partial_{\mathbf{x}}^2u, \dots, \partial_{\mathbf{x}}^k u, \dots) = c_1 + c_2 \partial_{x_1} u + \dots + c_m u \partial_{x_1} u + \dots. \quad (6.4)$$

where $\partial_{\mathbf{x}}^k$ denotes all k -th order partial derivatives and ∂_{x_i} denotes the partial derivative

with respect to the i -th variable. We refer to each term, such as $1, \partial_{x_1} u$, and $u \partial_{x_1} u, \dots$ in Equation 6.4, as a *feature*. Since f is a finite order polynomial, only a finite number of features are included. Denote the number of features by K . Under this model, the function f is expressed in a parametric form as a linear combination of K features. Our objective is to recover the parameters, or coefficients,

$$\mathbf{c} = [c_1 \ c_2 \ \dots \ c_m \ \dots \ c_K]^T \in \mathbb{R}^K,$$

where many of the entries may be zero.

From **D**, we numerically approximate the time and spatial derivatives of u to obtain the following *approximated time derivative vector* $D_t U \in \mathbb{R}^{NM^d}$ and *approximated feature matrix* $F \in \mathbb{R}^{NM^d \times K}$:

$$D_t U = \begin{bmatrix} \frac{U^1 - U^0}{\Delta t} \\ \frac{U^2 - U^1}{\Delta t} \\ \vdots \\ \frac{U^N - U^{N-1}}{\Delta t} \end{bmatrix}, \quad F = \begin{bmatrix} \mathbf{1}_{M^d \times 1} & U^0 & \dots & U^0 \circ D_{x_1} U^0 & \dots \\ \mathbf{1}_{M^d \times 1} & U^1 & \dots & U^1 \circ D_{x_1} U^1 & \dots \\ \vdots & \vdots & \ddots & \vdots & \dots \\ \mathbf{1}_{M^d \times 1} & U^{N-1} & \dots & U^{N-1} \circ D_{x_1} U^{N-1} & \dots \end{bmatrix}. \quad (6.5)$$

In this paper, the time derivatives in $D_t U$ are approximated by the forward difference scheme, and the spatial derivatives, such as $D_{x_1} U^n$ for $n = 0, 1, \dots, N-1$ in F are computed using the 5-point ENO scheme [387]. Our method can be applied if other numerical differentiation schemes are used. The vector $\mathbf{1}_{M^d \times 1} \in \mathbb{R}^{M^d}$ denotes the 1-vector of size M^d , and the Hadamard product \circ is the element-wise multiplication between two vectors. Each column of F is referred to as a *feature column*. The PDE model in Equation 6.2 suggests that, an optimal coefficient vector \mathbf{c} should satisfy the following approximation:

$$D_t U \approx F \mathbf{c}. \quad (6.6)$$

The objective is to find the correct set of coefficients in Equation 6.4. Due to the large size of K , the idea of sparsity becomes useful.

Throughout this chapter, we denote F_0 as the true feature matrix whose elements are the exact derivatives evaluated at the corresponding time and space location as those in F . For a vector \mathbf{c} , $\|\mathbf{c}\|_p := (\sum_j |c_j|^p)^{\frac{1}{p}}$ is the L_p norm of \mathbf{c} . In particular, $\|\mathbf{c}\|_\infty := \max_j |c_j|$. When $p = 0$, $\|\mathbf{c}\|_0 := \#\{c_j : c_j \neq 0\}$ represents the L_0 semi-norm of \mathbf{c} . The support of \mathbf{c} is denoted by $\text{supp}(\mathbf{c}) := \{j : c_j \neq 0\}$. The vector \mathbf{c} is said to be k -sparse if $\|\mathbf{c}\|_0 = k$ for a non-negative integer k . For any matrix $A_{m \times n}$ and index sets $\mathcal{L}_1 \subseteq \{1, 2, \dots, n\}$, $\mathcal{L}_2 \subseteq \{1, 2, \dots, m\}$, we denote $[A]_{\mathcal{L}_1}$ as the submatrix of A consisting of the columns indexed by \mathcal{L}_1 , and $[A]^{\mathcal{L}_2}$ as the submatrix of A consisting of the rows indexed by \mathcal{L}_2 . A^T , A^* and A^\dagger denote the transpose, conjugate transpose and Moore-Penrose pseudoinverse of A , respectively. For $x \in \mathbb{R}$, $\lfloor x \rfloor$ denotes the largest integer no larger than x . Moreover, we use the following notation for asymptotics: For sufficiently large n , we write $f(n) = \mathcal{O}(g(n))$, if there exists a constant $K > 0$ such that $f(n) \leq Kg(n)$, and $f(n) = \Omega(g(n))$ if $f(n) \geq K'g(n)$ for some constant $K' > 0$. The notation $f(n) = \Theta(g(n))$ means that $f(n) = \mathcal{O}(g(n))$ and $f(n) = \Omega(g(n))$. We adopt bold lower-case letters for vectors and bold upper-case letters for matrices. For a vector $\mathbf{v} \in \mathbb{R}^n$, $\|\mathbf{v}\|_1 := \sum_{i=1}^n |v_i|$, $\|\mathbf{v}\|_2 := \sqrt{\sum_{i=1}^n v_i^2}$, and $\|\mathbf{v}\|_\infty := \max_{1 \leq i \leq n} |v_i|$. For a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, \mathbf{A}^T denotes its transpose, $\|\mathbf{A}\|_2 := \max_{\|x\|_2=1} \|Ax\|_2$, $\|\mathbf{A}\|_\infty := \max_{1 \leq i \leq n} \sum_{j=1}^m |A_{i,j}|$, $\|\mathbf{A}\|_{\infty, \infty} := \max_{1 \leq i, j \leq n} |A_{i,j}|$, and $\|\mathbf{A}\|_F := \sqrt{\sum_{i=1}^n \sum_{j=1}^m A_{i,j}^2}$.

6.1.2 Noise Amplification during Differentiation

Despite the developments of many useful methods, when the given data are noisy, PDE identification is still challenging. A small amount of noise can make a recovery unstable, especially for high order PDEs. It was shown in [369] that the noise to signal ratio for LASSO depends on the order of the underlying PDE, and IDENT can handle a small amount of noise when the PDE contains high order derivatives. A significant issue is that

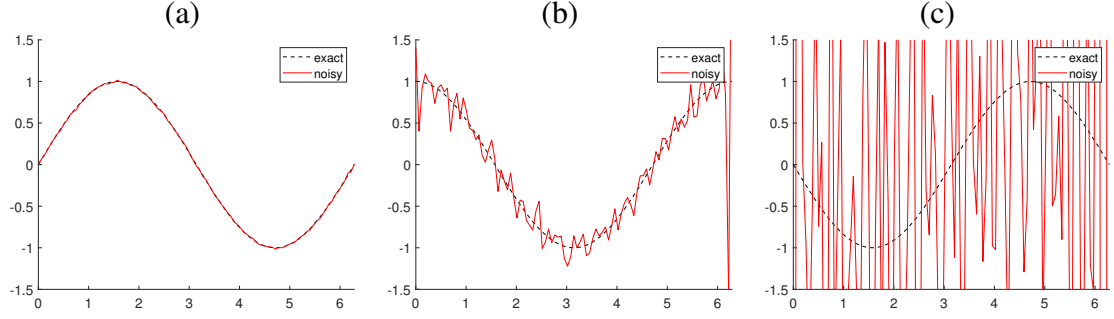


Figure 6.1: The sensitivity of numerical differentiation to noise. (a) Graph of $\sin(x)$, $0 \leq x \leq 2\pi$ (black), and its noisy version (red) with Gaussian noise of mean 0 and standard deviation 0.01. (b) The first-order derivatives of the function (black) and the data (red). (c) The second-order derivatives of the function (black) and the data (red). The derivatives of data in (b) and (c) are computed using the five-point ENO scheme. As the order of derivative increases, the noise gets amplified.

the numerical differentiation often magnifies noise, which is illustrated by an example in Figure 6.1.

In the following, we introduce a class of robust PDE identification methods that can handle a large amount of noise.

6.1.3 Successively Denoised Differentiation (SDD)

As shown in Figure 6.1, when the given data are contaminated by noise, numerical differentiation amplifies noise. It introduces a large error in the time derivative vector $D_t U$ and the approximated feature matrix F . With random noise, the regularity of the given data is different from the PDE solution's regularity. Thus, the denoising plays a vital role in PDE identification.

We introduce a smoothing operator S to process the data. Kernel methods are good options for S , such as Moving Average [388] and Moving Least Square (MLS) [389]. In this paper, the smoothing operator S is chosen as the MLS, where data are locally fit by quadratic polynomials. In MLS, a weighted least squares problem, in the time domain or

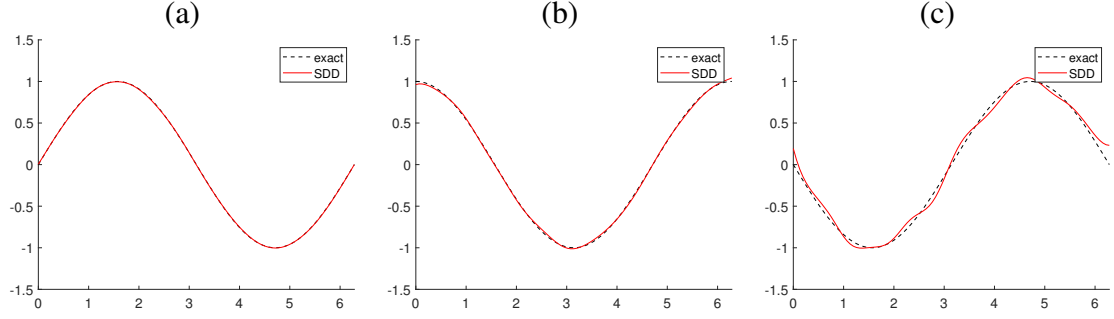


Figure 6.2: Performance of SDD on the data in Figure 6.1. (a) Graph of $\sin(x)$, $0 \leq x \leq 2\pi$ (black) and the denoised data (red) using MLS. (b) First-order derivatives of the function (black) and the denoised data using SDD (red). (c) Second-order derivatives of the function (black) and the denoised data using SDD (red). Derivatives are computed by the five-point ENO scheme, and the smoothing operator S is MLS.

the spatial domain, is solved at each time t^n and spatial location x_i as follows:

$$S_{(x)}[U_i^n] = p_i^n(\mathbf{x}_i), \text{ with } p_i^n = \arg \min_{p \in P_2} \sum_{\mathbf{j} \in \mathbb{I}} (p(\mathbf{x}_j) - U_j^n)^2 \exp \left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{h^2} \right), \quad (6.7)$$

$$S_{(t)}[U_i^n] = p_i^n(t^n), \text{ with } p_i^n = \arg \min_{p \in P_2} \sum_{0 \leq k \leq N} (p(t^k) - U_i^k)^2 \exp \left(-\frac{\|t^n - t^k\|^2}{h^2} \right). \quad (6.8)$$

Here $h > 0$ is a width parameter of the kernel, and P_2 denotes the set of polynomials of degree no more than 2. When data are one dimensional, i.e. $d = 1$ and for a fixed time index n , we say that the given data set $\{U_i^n\}_{i=0}^{M-1}$ with the spatial grid length Δx is a third order approximation of a smooth function $u(x, t^n)$, if $|U_i^n - u(x_i, t^n)| = O(\Delta x^3)$ for any i . It has been shown that if $\{U_i^n\}_{i=0}^{M-1}$ is a third-order approximation of a smooth function $u(x, t^n)$, and if h is properly chosen, the output of MLS also gives a third-order approximation of $u(x, t^n)$ [390, 369]. Such a result also holds for a fixed spatial index i when we consider the third-order approximation of $u(x_i, t)$ in terms of the time grid length Δt . This result can be generalized to multi-dimensions and higher-order polynomial approximations in MLS. In practice, the width parameter h is found empirically from data.

We describe the Successively Denoised Differentiation (SDD) procedure to stabilize the numerical differentiation [386]. For every derivative approximation, smoothing is applied

Table 6.1: The procedure of SDD, where the spatial and time smoothing operators $S_{(\mathbf{x})}$ and $S_{(t)}$ are defined in Equation 6.7 and Equation 6.8 respectively. The operator D_t given in Equation 6.5 represents numerical time differentiation by the forward difference scheme, and $D_{x_i}U$ for $i = 0, 1, \dots, N - 1$ represents numerical spatial differentiation with respect to x_i given by the 5-point ENO scheme [387].

Successively Denoised Differentiation (SDD)	
Expression for Approximation	Explanation
$u \approx S_{(\mathbf{x})}[U]$	The given data set U is denoised by MLS.
$\partial_t u \approx S_{(t)} D_t S_{(\mathbf{x})}[U]$	Denoising at numerical time differentiation.
$\partial_{\mathbf{x}}^{\mathbf{k}} u \approx (S_{(\mathbf{x})} D_{x_1})^{k_1} \dots (S_{(\mathbf{x})} D_{x_d})^{k_d} S_{(\mathbf{x})}[U],$ where $\mathbf{k} = (k_1, \dots, k_d)$, and $\sum_{i=1}^d k_i = k$ for $k = 1, 2, \dots$	Denoising at every step of the numerical spatial differentiation.

as described in Table 6.1. The main idea of SDD is to smooth the data at each step (before and after) the numerical differentiation. This simple idea effectively stabilizes numerical differentiation. Figure 6.2 shows the results of SDD for the same data in Figure 6.1. The approximations of the first and second-order derivatives of u are greatly improved.

In subsection 6.3.8, we explore details of SDD when different smoothing operators are used. We find that MLS has the best performance in terms of preserving the derivative profiles. Therefore, we set S to be MLS in our numerical experiments.

To simplify the notations, in the rest of this paper, we use U to denote the denoised data $S_{(\mathbf{x})}[U]$, and $D_t U$ as well as $D_{\mathbf{x}}^k U$ to denote the numerical derivatives with SDD applied as above.

6.2 PDE Model Identification Methods: ST and SC

Under the parametric model in Equation 6.4, the PDE identification problem can be reduced to solving the linear system (Equation 6.6) for a sparse vector \mathbf{c} with few nonzero entries.

Sparse regression can be formulated as the following L_0 -minimization

$$\min \|\mathbf{c}\|_0, \quad \text{subject to } \|F\mathbf{c} - D_t U\| \leq \epsilon, \quad (6.9)$$

for some $\epsilon > 0$. However, the L_0 -minimization in Equation 6.9 is NP-hard. Its approximate solutions have been intensively studied in the literature. The most popular surrogate for the L_0 semi-norm is the L_1 norm as applied in image and signal processing [391, 392]. The L_1 -regularized minimization is called Least Absolute Shrinkage and Selection Operator (LASSO) [393], which was used in [369, 367, 368] for PDE identification. The common strategy in these works is to utilize LASSO to select a candidate set, then refine the results with other techniques.

We utilize a greedy algorithm called Subspace Pursuit (SP) [394] to select a candidate set. Unlike LASSO, SP takes the sparsity as an input, allowing direct control of the sparsity of the reconstructed coefficient. Let k be a positive integer and denote $\mathbf{b} = D_t U$. For a fixed sparsity level k , $\text{SP}(k; F, \mathbf{b})$ in algorithm 9 gives rise to a k -sparse vector whose support is selected in a greedy fashion. It was proved that SP gives rise to a solution of the L_0 -minimization (Equation 6.9) under certain conditions of the matrix F , such as the restricted isometry property [394].

We introduce two new methods based on SP for PDE identification: Subspace pursuit Time evolution (ST) and Subspace pursuit Cross-validation (SC). ST uses multi-shooting numerical time evolution and selects the PDE, which yields the least evolution error. From a different perspective, SC computes the cross-validation error in the least-squares fitting and picks the PDE that gives the smallest error.

6.2.1 Subspace Pursuit Time Evolution (ST)

We first describe a method combining SP and the idea of time evolution. In [369], Time Evolution Error (TEE) quantifies the mismatch between the solution simulated from a can-

Input: $F \in \mathbb{R}^{NM^d \times K}$, $\mathbf{b} \in \mathbb{R}^{NM^d}$ and sparsity $k \in \mathbb{N}$.

Initialization: $j = 0$;
 $G \leftarrow$ column-normalized version of F ;
 $\mathcal{I}^0 = \{k \text{ indices corresponding to the largest magnitude entries in the vector } G^* \mathbf{b}\}$;
 $\mathbf{b}_{\text{res}}^0 = \mathbf{b} - G_{\mathcal{I}^0} G_{\mathcal{I}^0}^\dagger \mathbf{b}$.

while *True* **do**

Step 1. $\tilde{\mathcal{I}}^{j+1} = \mathcal{I}^j \cup \{k \text{ indices corresponding to the largest magnitude entries in the vector } G^* \mathbf{b}_{\text{res}}^j\}$;

Step 2. Set $\mathbf{c}_p = G_{\tilde{\mathcal{I}}^{j+1}}^\dagger \mathbf{b}$;

Step 3. $\mathcal{I}^{j+1} = \{k \text{ indices corresponding to the largest elements of } \mathbf{c}_p\}$;

Step 4. Compute $\mathbf{b}_{\text{res}}^{j+1} = \mathbf{b} - G_{\mathcal{I}^{j+1}} G_{\mathcal{I}^{j+1}}^\dagger \mathbf{b}$;

Step 5. If $\|\mathbf{b}_{\text{res}}^{j+1}\|_2 > \|\mathbf{b}_{\text{res}}^j\|_2$, let $\mathcal{I}^{j+1} = \mathcal{I}^j$ and terminate the algorithm; otherwise set $j \leftarrow j + 1$ and iterate.

end

Output: $\hat{\mathbf{c}} \in \mathbb{R}^K$ satisfying $\hat{\mathbf{c}}_{\mathcal{I}_j} = F_{\mathcal{I}_j}^\dagger \mathbf{b}$ and $\hat{\mathbf{c}}_{(\mathcal{I}_j)^c} = \mathbf{0}$.

Algorithm 9: Subspace Pursuit SP($k; F, \mathbf{b}$)

didate PDE and the denoised data. Any candidate coefficient vector $\hat{\mathbf{c}} = (\hat{c}_1, \hat{c}_2 \dots)$ defines a candidate PDE:

$$u_t = \hat{c}_1 + \hat{c}_2 \partial_{x_1} u + \dots + \hat{c}_m u \partial_{x_1} u + \dots$$

This PDE is numerically evolved from the initial condition U^0 with a smaller time step $\widetilde{\Delta t} \ll \Delta t$. Specifically, if r is the highest order of the spatial derivatives, we set the time step as $c(\Delta x)^r$ with some constant $c < 1$. Denote $\hat{U}^1, \hat{U}^2, \dots, \hat{U}^N$ as this numerical solution at the same time-space location as U^1, U^2, \dots, U^N . The TEE of the candidate PDE given by $\hat{\mathbf{c}}$ is

$$\text{TEE}(\hat{\mathbf{c}}) = \frac{1}{N} \sum_{n=1}^N \|\hat{U}^n - U^n\|_2,$$

where U^n is the denoised data at time t^n . Figure 6.3 (a) and (b) illustrate the idea of TEE.

When there are several candidate PDEs, the one with the least TEE is picked [369]. This TEE idea is based on the convergence principle that a correct numerical approximation converges to the true solution as the time step $\widetilde{\Delta t}$ goes to zero. The error from the wrongly identified terms grows during this time evolution process [369]. More specifically, assume

that the solution u is smooth and decays sufficiently fast at infinity. Consider the following linear equation with constant coefficients:

$$\frac{\partial u}{\partial t} = a_0 u + a_1 \frac{\partial u}{\partial x} + \cdots + a_m \frac{\partial^m u}{\partial x^m}.$$

After taking the Fourier transform for the equation and solving the ODE, one can obtain the transformed solution:

$$\hat{u}(\xi, t) = \hat{u}(\xi, 0) e^{a_0 t} e^{a_1 i \xi t} e^{-a_2 \xi^2 t} \cdots e^{a_m (i \xi)^m t},$$

where $i = \sqrt{-1}$ and ξ is the variable in the Fourier domain. If a term with an even-order derivative, such as $a_2 \frac{\partial^2 u}{\partial x^2}$, is mistakenly included in the PDE, it will make every frequency mode grow or decrease exponentially in time; if a term with an odd-order derivative, such as $a_1 \frac{\partial u}{\partial x}$, is mistakenly included in the solution, it will introduce a wrong-speed oscillation of the solution. In either case, the correct solution's deviation grows fast in time, providing an efficient way to distinguish the wrong terms.

We introduce a *Multi-shooting Time Evolution Error* (MTEE). The idea is to evolve a candidate PDE from multiple time locations with a time step $\widetilde{\Delta t} \ll \Delta t$ using the forward Euler scheme for a time length of $w \Delta t$, where w is a positive integer. Let $\widehat{U}^{(n+w)|n}$ be the numerical solution of the candidate PDE at the time $(n+w)\Delta t$, which is evolved from the initial condition U^n at time $t^n = n\Delta t$. The MTEE is defined as

$$\text{MTEE}(\widehat{\mathbf{c}}; w) = \frac{1}{N-w} \sum_{n=0}^{N-1-w} \|\widehat{U}^{(n+w)|n} - U^{n+w}\|_2. \quad (6.10)$$

Figure 6.3 (c) and (d) demonstrate the process of multi-shooting time evolution. While the TEE evolution starts from the initial condition U^0 and ends at T , the MTEE evolution starts from various time locations, such as $t^n, n = 0, \dots, N-1-w$, and lasts for a shorter time, e.g., $w\Delta t$ in our case.

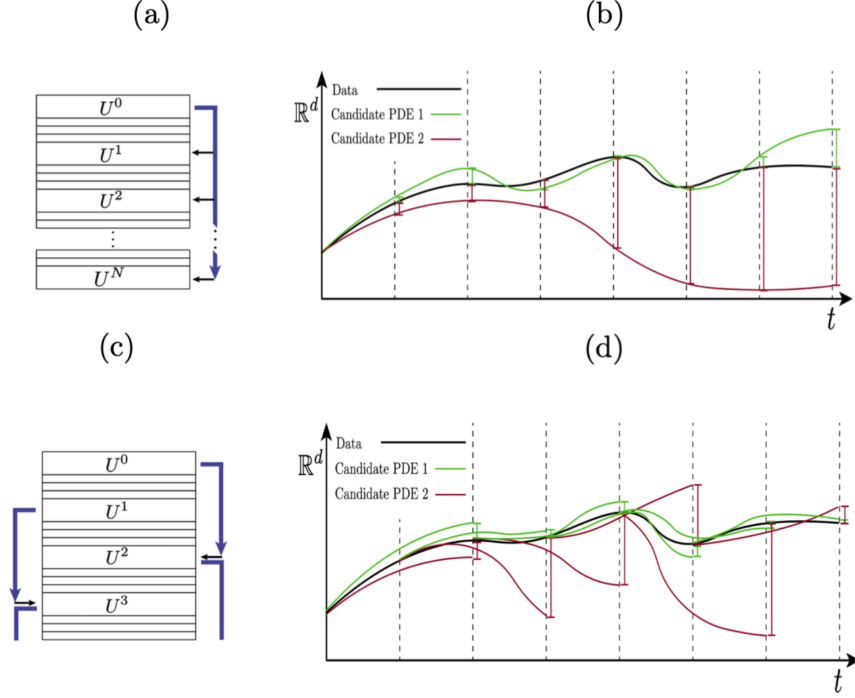


Figure 6.3: (a) and (b) illustrate the idea of TEE. (c) and (d) explain MTEE when $w = 2$. The blue arrows in (a) and (c) represent time evolution using the forward Euler scheme on a fine time grid with spacing $\widetilde{\Delta t} \ll \Delta t$. In (b), two different PDEs (green and red) are evolved, and the green one has a smaller TEE. In (d), the candidate PDEs are evolved from multiple time locations, and their numerical solutions are compared with the denoised data after a time length of $w\Delta t$.

MTEE has two advantages over TEE: (1) MTEE is more robust against noise in comparison with TEE. If $w \ll N$, the noise in the initial condition accumulates for a smaller amount of time in MTEE, which helps to stabilize numerical solvers.

For example, given a set of noisy solution of the Burgers' equation $u_t = -uu_x$ generated with $T = 0.05$, $\Delta t = 0.001$, $\Delta x = 1/256$ (as shown in Figure 6.5(a)), if we evolve the noisy initial condition with the correct PDE, i.e., $u_t = -uu_x$, the solution blows up at $t = 0.032$ (the solution before the blow up at $t = 0.03$ is shown in Figure 6.5(c)). As a result, the TEE does not exist and cannot be used. On the other hand, if we evolve the initial condition for a shorter time, say $t = 0.02$, we can get a solution (as shown in Figure 6.5(b)). Thus MTEE can be computed and used for PDE identification.

(2) MTEE is more flexible, and its computation is parallelizable. The flexibility of

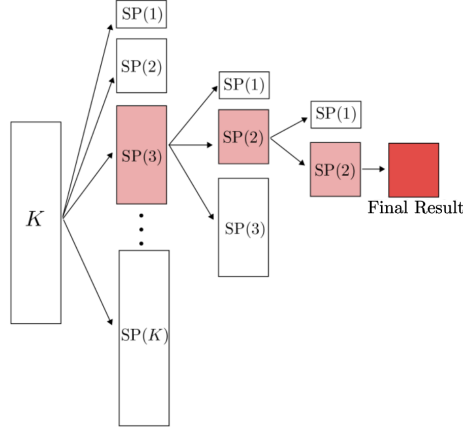


Figure 6.4: An example of the ST iteration. Starting with a large number K , the first iteration gives rise to K candidate coefficients for $k = 1, \dots, K$. The PDE with the smallest MTEE is picked, e.g., SP(3) with cardinality $K_1 = 3$ and support \mathcal{A}_1 . The second iteration gives rise to the candidate coefficients only supported on \mathcal{A}_1 using SP(k) with $k = 1, 2, 3$. The PDE with the smallest MTEE is found, e.g., SP(2) with cardinality $K_2 = 2$ and support \mathcal{A}_2 . The third iteration does not change the support, i.e., $\mathcal{A}_3 = \mathcal{A}_2$, so the final output is the coefficient vector of SP(2).

MTEE comes from two aspects: (1) The error accumulation time can be controlled by the parameter w such that the PDE is evolved for a time length of $w\Delta t$. (2) One may assign different weights in the calculation of the evolution errors in different periods. Since each time evolution in the multi-shooting is independent, the computation of MTEE can be parallelized.

The SP algorithm finds a coefficient vector with a specified sparsity, but it is difficult to know the correct sparsity from the given data. We propose *Subspace pursuit Time evolution* (ST), which iteratively refines the selection of features.

As an initial condition, we set $K_0 = K$ and $\mathcal{A}_0 = \{1, \dots, K\}$. At the first iteration, all possible sparsity levels are considered in the SP algorithm. For each $k = 1, \dots, K$, we run $\text{SP}(k; F, D_t U)$ to obtain a coefficient vector $\hat{\mathbf{c}}^{(k)} \in \mathbb{R}^K$ such that $\|\hat{\mathbf{c}}^{(k)}\|_0 = k$, which gives rise to the PDE:

$$u_t = f_{\text{SP}(k)} \text{ where } f_{\text{SP}(k)} := \hat{c}_1^{(k)} + \hat{c}_2^{(k)} \partial_{x_1} u + \dots + \hat{c}_m^{(k)} u \partial_{x_1} u + \dots \quad (6.11)$$

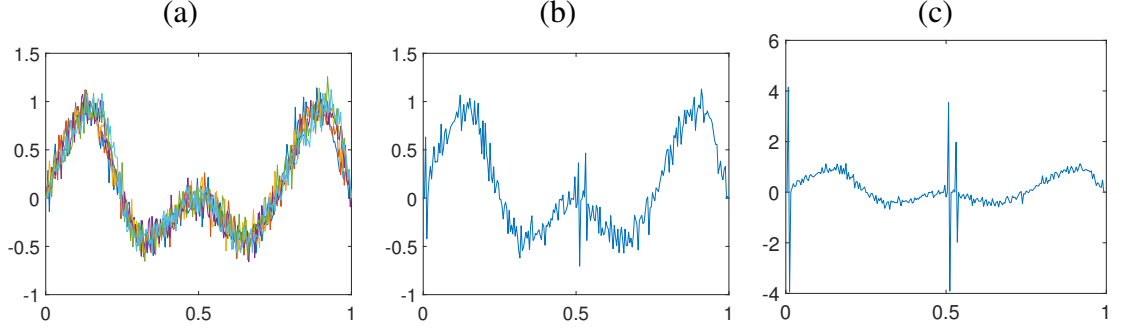


Figure 6.5: Robustness of MTEE over TEE. (a) Noisy solution set of the Burgers' equation $u_t = -uu_x$ generated with $T = 0.05$, $\Delta t = 0.001$, $\Delta x = 1/256$. By evolving the noisy initial condition according to $u_t = -uu_x$, (b) shows the solution at $t = 0.02$ and (c) shows the solution at $t = 0.03$. The solution blows up at $t = 0.032$.

Input: $F \in \mathbb{R}^{NM^d \times K}$, $D_t U \in \mathbb{R}^{NM^d}$ and a positive integer w .

Initialization: $j = 0$, $K_0 = K$ and $\mathcal{A}_0 = \{1, 2, \dots, K\}$.

while $\mathcal{A}_{j+1} \neq \mathcal{A}_j$ **do**

Step 1. For $k = 1, 2, \dots, K_j$, run $\text{SP}(k; [F]_{\mathcal{A}_j}, D_t U)$ to obtain a coefficient vector $\hat{\mathbf{c}}^{(k)} \in \mathbb{R}^K$ such that

$$\hat{\mathbf{c}}_{\mathcal{A}_j}^{(k)} = \text{SP}(k; [F]_{\mathcal{A}_j}, D_t U) \text{ and } \hat{\mathbf{c}}_{\mathcal{A}_j^c}^{(k)} = \mathbf{0},$$

and the associated PDE $u_t = f_{\text{SP}(k)}$ given in Equation 6.11.

Step 2. Among all the PDEs $u_t = f_{\text{SP}(k)}$ for $k = 1, \dots, K_j$, select the one with the minimum $\text{MTEE}(\hat{\mathbf{c}}^{(k)}; w)$ and update

$$K_{j+1} = \arg \min_{k=1,2,\dots,K_j} \text{MTEE}(\hat{\mathbf{c}}^{(k)}; w) \text{ and } \mathcal{A}_{j+1} = \text{supp}(\hat{\mathbf{c}}^{(k_{j+1})}).$$

If $\mathcal{A}_{j+1} = \mathcal{A}_j$, terminate the algorithm; otherwise, update $j = j + 1$.

end

Output: Recovered coefficient $\hat{\mathbf{c}}^{K_{j+1}}$ and the corresponding PDE, denoted by $\text{ST}(w)$.

Algorithm 10: Subspace pursuit Time evolution (ST)

We then numerically evolve each PDE $u_t = f_{\text{SP}(k)}$, for $k = 1, \dots, K$ and calculate the corresponding MTEE. Among these PDEs, the one with the smallest MTEE is selected, then let

$$K_1 = \arg \min_{k=1,2,\dots,K} \text{MTEE}(\hat{\mathbf{c}}^{(k)}; w) \text{ and } \mathcal{A}_1 = \text{supp}(\hat{\mathbf{c}}^{(K_1)}) .$$

If $\mathcal{A}_1 = \mathcal{A}_0$, the algorithm is terminated; otherwise, we continue to the second iteration. The proposed method requires solving the sparsity-constrained least-squares problems at least K times. Meanwhile, these computations and the evaluation of MTEE are parallelizable.

At the second iteration, we refine the selection from the index set \mathcal{A}_1 with cardinality K_1 . For $k = 0, \dots, K_1$, we run $\text{SP}(k; [F]_{\mathcal{A}_1}, D_t U)$ to obtain a coefficient vector $\hat{\mathbf{c}}^{(k)} \in \mathbb{R}^K$ such that

$$\hat{\mathbf{c}}_{\mathcal{A}_1}^{(k)} = \text{SP}(k; [F]_{\mathcal{A}_1}, D_t U) , \text{ and } \hat{\mathbf{c}}_{\mathcal{A}_1^c}^{(k)} = \mathbf{0} ,$$

and the associated PDE $u_t = f_{\text{SP}(k)}$ as in (Equation 6.11). Among these PDEs, the one with the smallest MTEE is selected, and we denote

$$K_2 = \arg \min_{k=1,2,\dots,K_1} \text{MTEE}(\hat{\mathbf{c}}^{(k)}; w) , \text{ and } \mathcal{A}_2 = \text{supp}(\hat{\mathbf{c}}^{(K_2)}) .$$

If $\mathcal{A}_2 = \mathcal{A}_1$, the algorithm is terminated; otherwise, we continue to the next iteration similarly.

The ST iteration will be terminated when the index set remains the same, i.e., $\mathcal{A}_j = \mathcal{A}_{j+1}$. The ST outputs a recovered coefficient vector and the corresponding PDE denoted by $\text{ST}(w)$. A complete description of ST is given in Algorithm algorithm 10, and Figure 6.4 illustrates an example of the ST iteration.

6.2.2 Subspace Pursuit Cross Validation (SC)

Our second method utilizes the idea of cross-validation for the linear system in Equation 6.6. Cross-validation is commonly used in statistics for the choice of parameters in

order to avoid overfitting [395]. We consider the two-fold cross-validation where data are partitioned into two subsets. One subset is used to estimate the coefficient vector, and the other one is used to validate the candidates. If a suitable coefficient vector is found within one subset, it should yield a small validation error for the other subset because of consistency.

For some fixed ratio parameter $\alpha \in (0, 1)$, we split the rows of $D_t U \in \mathbb{R}^{NM^d}$ (and $F \in \mathbb{R}^{NM^d \times K}$) into two groups indexed by \mathcal{T}_1 and \mathcal{T}_2 , such that \mathcal{T}_1 consists of the indices of the first $\lfloor \alpha NM^d \rfloor$ rows and \mathcal{T}_2 consists of the indices of the rest of the rows. Since we focus on PDEs with constant coefficients, the idea of cross validation is applicable: if a correct support is identified, the coefficient vector obtained from the data in \mathcal{T}_1 should be compatible with the data in \mathcal{T}_2 .

We introduce our *Subspace pursuit Cross-validation (SC)* algorithm where cross-validation is incorporated into the SP algorithm. SC consists of the following three steps:

Step 1: For each sparsity level $k = 1, 2, \dots, K$, use SP to select a set of active features:

$$\mathcal{A}_k = \text{supp}(\text{SP}(k; F, D_t U)) .$$

Step 2: Use the data in \mathcal{T}_1 to compute the estimator for the coefficient vector, $\hat{\mathbf{c}}^{(k)} \in \mathbb{R}^K$, by the following least squares problem

$$\hat{\mathbf{c}}^{(k)} = \arg \min_{\mathbf{c} \in \mathbb{R}^K \text{ such that } \mathbf{c}_{\mathcal{A}_k^c} = 0} \|[F]_{\mathcal{A}_k}^{\mathcal{T}_1} \mathbf{c}_{\mathcal{A}_k} - [D_t U]^{\mathcal{T}_1}\|_2^2 ,$$

and then use the data in \mathcal{T}_2 to compute a Cross-validation Estimation Error (CEE)

$$\text{CEE}(\mathcal{A}_k; \alpha, \mathcal{T}_1, \mathcal{T}_2) = \|[D_t U]^{\mathcal{T}_2} - [F]^{\mathcal{T}_2} \hat{\mathbf{c}}^{(k)}\|_2 . \quad (6.12)$$

Step 3: Set $k_{\min} = \arg \min_k \text{CEE}(\mathcal{A}_k; \alpha, \mathcal{T}_1, \mathcal{T}_2)$ and the estimated coefficient vector is

given as

$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathbb{R}^K \text{ such that } \mathbf{c}_{\mathcal{A}_k^c} = 0} \|[F]_{\mathcal{A}_{k_{\min}}^{\mathcal{T}_1}} \mathbf{c}_{\mathcal{A}_{k_{\min}}} - [D_t U]^{\mathcal{T}_1}\|_2^2.$$

The identified PDE by SC is denoted as SC(α).

CEE in Equation 6.12 is an effective measure for consistency. If the estimated coefficient vector's support matches that of the true one, CEE is guaranteed to be small provided with sufficiently high resolution in time and space.

Theorem 6.2.1. *Assume that $D_t U \rightarrow u_t$ and $F \rightarrow F_0$ pointwise as $\Delta t, \Delta x \rightarrow 0$. Let $\mathcal{A}_0 = \text{supp}(\mathbf{c}_0)$ where \mathbf{c}_0 is the coefficient vector of the true PDE. For any set of support \mathcal{A} , we have*

$$\text{CEE}(\mathcal{A}; \alpha, \mathcal{T}_1, \mathcal{T}_2) \leq \left\| \left([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger \right) [u_t]^{\mathcal{T}_1} \right\|_2 + g(\mathcal{A}; \alpha, \mathcal{T}_1, \mathcal{T}_2),$$

where $g > 0$ is a summation of residual terms of approximating the partial derivatives and feature matrix using data, see Equation C.1, which is independent of \mathcal{A}_0 , such that $g \rightarrow 0$ as $\Delta t, \Delta x \rightarrow 0$.

Proof. See section C.1. □

In Equation 6.12, the data in \mathcal{T}_1 serve as the training set, and the data in \mathcal{T}_2 act as the validation set. One can also use the data in \mathcal{T}_2 for training and the data in \mathcal{T}_1 for validation, which gives rise to the cross validation estimation error $\text{CEE}(\mathcal{A}_k; 1 - \alpha, \mathcal{T}_2, \mathcal{T}_1)$. To improve the robustness of SC, we replace Equation 6.12 with the following averaged cross-validation error:

$$\text{CEE}(\mathcal{A}_k, \alpha) = \frac{1}{2} (\text{CEE}(\mathcal{A}_k; \alpha, \mathcal{T}_1, \mathcal{T}_2) + \text{CEE}(\mathcal{A}_k; 1 - \alpha, \mathcal{T}_2, \mathcal{T}_1)).$$

In general, one can randomly pick a part of the data as the training set and use the rest

as the validation set. For simplicity, we split the data according to the row index in this paper.

The proposed SC algorithm is summarized in algorithm 11. In comparison with ST, SC does not involve any numerical evolution of the candidate PDE, so the computation of SC is faster.

Input: $F \in \mathbb{R}^{NM^d \times K}$ and $D_t U \in \mathbb{R}^{NM^d}$; $0 < \alpha < 1$ ratio of the training data.

Step 1. For $k = 1, 2, \dots, K$, run $\text{SP}(k; F, D_t U)$ to obtain the support of the candidate coefficients

$$\mathcal{A}_k = \text{supp}(\text{SP}(k; F, D_t U)) .$$

Step 2. For each k , compute the averaged cross validation error

$$\text{CEE}(\mathcal{A}_k, \alpha) = \frac{1}{2} (\text{CEE}(\mathcal{A}_k; \alpha, \mathcal{T}_1, \mathcal{T}_2) + \text{CEE}(\mathcal{A}_k; 1 - \alpha, \mathcal{T}_2, \mathcal{T}_1)) .$$

Step 3. Choose the k which gives the smallest cross validation error and denote it by k_{\min}

$$k_{\min} = \arg \min_k \text{CEE}(\mathcal{A}_k, \alpha) .$$

Estimate the coefficients by least squares as

$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in \mathbb{R}^K \text{ such that } \mathbf{c}_{\mathcal{A}_k^c} = 0} \|[F]_{\mathcal{A}_{k_{\min}}}^{\mathcal{T}_1} \mathbf{c}_{\mathcal{A}_{k_{\min}}} - [D_t U]^{\mathcal{T}_1}\|_2^2 .$$

Output: Recovered coefficient $\hat{\mathbf{c}}$ and the identified PDE denoted by $\text{SC}(\alpha)$.

Algorithm 11: Subspace pursuit Cross validation (SC) Algorithm

6.3 Numerical Experiments on Robust PDE Identification

In this section, we perform a systematic numerical study to demonstrate the effectiveness of ST and SC and compare them to IDENT [369]. To measure the identification error, we

use the following relative coefficient error e_c and grid-dependent residual error e_r :

$$e_c = \frac{\|\hat{\mathbf{c}} - \mathbf{c}\|_1}{\|\mathbf{c}\|_1}, \quad e_r = \begin{cases} \sqrt{\Delta x \Delta t} \|F(\hat{\mathbf{c}} - \mathbf{c})\|_2 & \text{for 1D PDE.} \\ \sqrt{\Delta x \Delta y \Delta t} \|F(\hat{\mathbf{c}} - \mathbf{c})\|_2 & \text{for 2D PDE.} \end{cases} \quad (6.13)$$

The relative coefficient error e_c measures the accuracy in the recovery of PDE coefficients, while the residual error e_r measures the difference between the learned dynamics and the denoised one by SDD. Since each feature vector in F may have different scales, e_r can be different from e_c in some cases. When the given data contain noise, the features containing higher-order derivatives have greater magnitudes than the features containing lower order derivatives. In this case, a small coefficient error in the high order terms may lead to a large e_r . We use both e_c and e_r to quantify the PDE identification error.

To measure how well the solution of the identified PDE matches the dynamics of the correct PDE, we use the following evolution error

$$e_e = \Delta x \Delta t \left(\sum_n \sum_{\mathbf{i}} |u(\mathbf{x}_{\mathbf{i}}, t^n) - \hat{u}(\mathbf{x}_{\mathbf{i}}, t^n)| \right) \quad (6.14)$$

where u and \hat{u} denote the solution of the exact and identified PDE respectively.

To generate the data, we first solve the underlying PDE by forward Euler scheme using time and space step δt and δx (and δy) respectively, then downsample the data with time and space step Δt and Δx (and Δy). In the noisy case, we add Gaussian noise with standard deviation σ to the clean data. We say that the noise is $p\%$ by setting $\sigma = \frac{p}{100} \sqrt{\frac{1}{NM^d} \sum_n \sum_{\mathbf{i}} (u(\mathbf{x}_{\mathbf{i}}, t^n))^2}$. In the computation of $D_t U$ and the feature matrix F , we always use SDD with MLS with $h = 0.04$ as the smoother. When MLS is used to denoise the data of two dimensional PDEs, one can either fit two-dimensional polynomials or fit one-dimensional polynomials in each dimension. In this work, we use the second approach. In ST, without specification, $\widetilde{\Delta t} = \Delta t/5$ is used.

We first consider PDEs containing partial derivatives up to the second order. Let

the governing equation f be a polynomial with degree up to 2. There are 10 features: $1, u, u^2, u_x, u_x^2, uu_x, u_{xx}, u_{xx}^2, uu_{xx}, u_x u_{xx}$ in the dictionary for one dimensional PDEs. For two dimensional PDEs, there are 28 features, which contain $1, u, u_x, u_y, u_{xx}, y_{xy}, u_{yy}$ and their pairwise products. In the following examples, the spatial domain $[0, 1]$ is used for one-dimensional PDEs and $[0, 1]^2$ is used for two-dimensional PDEs. For both cases, zero Dirichlet boundary condition is used for all examples.

6.3.1 Transport Equation

Our first experiment is a transport equation with zero Dirichlet boundary condition:

$$u_t = -u_x, \quad (6.15)$$

with an initial condition of

$$u(x, 0) = \begin{cases} \sin^2(2\pi x/(1 - T)) \cos(2\pi x/(1 - T)), & \text{for } 0 \leq x \leq 1 - T, \\ 0, & \text{otherwise} \end{cases},$$

for $0 \leq x \leq 1$ and $0 < t \leq T$. The clean data \mathbf{D} is generated by explicitly solving Equation 6.15 with $\delta x = \Delta x = 1/256, \delta t = \Delta t = 10^{-3}$ and $T = 0.05$. In theory, for the transport equation, the zero boundary condition should only be applied to the inflow boundary. We design our initial condition and choose the evolution time T in such a way that the evolution of the solution is only restricted within the given spatial domain so that the PDE is well-defined. The same setup is considered in the rest of this section.

Table 6.2 shows the results of ST(20) and SC(1/200) with various noise levels. In practice, we have no a priori knowledge of whether the given data contain noise, so we conduct two experiments with and without SDD to check the effect of SDD on clean data. We observe that SDD makes a small difference in the noise-free case. With clean data, SC identifies an additional u_{xx} term with a small coefficient, while ST can rule out all wrong

Table 6.2: Identification of the transport equation (Equation 6.15) with different noise levels. In the noise-free case, applying SDD does not introduce a strong bias. The identification results (second column) by ST and SC are stable even with 30% noise. Here $w = 20$ for ST, and $\alpha = 1/200$ for SC.

Method	0% noise without SDD	e_c	e_r
ST	$u_t = -0.9994u_x$	6.20×10^{-4}	4.89×10^{-4}
SC	$u_t = -0.9993u_x - 0.0010u_{xx}$	1.65×10^{-3}	1.11×10^{-2}
	0% noise with SDD	e_c	e_r
ST	$u_t = -0.9997u_x$	3.36×10^{-4}	2.64×10^{-4}
SC	$u_t = -0.9997u_x - 0.0010u_{xx}$	1.34×10^{-3}	1.11×10^{-2}
	10% noise without SDD	e_c	e_r
ST	$u_t = -3.028 \times 10^{-4}u_{xx}$	1.00	5.55
SC	$u_t = 9.4224u - 2.9992u_{xx}$	1.04×10	5.62
	10% noise with SDD	e_c	e_r
ST, SC	$u_t = -1.0357u_x$	3.57×10^{-2}	2.67×10^{-2}
	30% noise without SDD	e_c	e_r
ST	$u_t = 8.0587 \times 10u - 2.6316 \times 10^{-4}u_{xx}$	8.16×10	1.88×10
SC	$u_t = 8.2488 \times 10u$	8.25×10	1.86×10
	30% noise with SDD	e_c	e_r
ST, SC	$u_t = -0.9421u_x$	5.79×10^{-2}	4.31×10^{-2}

terms. The corresponding e_c and e_r are both small. For 10% or 30% noise, the results by ST and SC with and without SDD are also shown. With SDD, both ST and SC identify the correct PDE with small e_c and e_r values. SDD significantly improves the results.

To further demonstrate the significance of SDD and the effectiveness of ST and SC, we display the noisy data with 10% and 30% noise, the denoised data, and the recovered dynamics in Figure 6.6. Even though the given data contain a large amount of noise, the recovered dynamics are close to the clean data. In the rest examples, SDD is always used for ST and SC on noisy data.

Figure 6.7 shows how e_c , e_r and e_e change when the noise level varies. Each experiment is repeated 50 times and the error is averaged. We test IDENT, ST(20) and SC(1/200). Figure 6.7 (a) shows that e_c of ST or SC is much smaller than that of IDENT when the noise level is larger than 20%. Figure 6.7 (b) and (c) shows e_r and e_e versus noise, respectively. The coefficient error e_c by ST and SC is significantly smaller than that of IDENT.

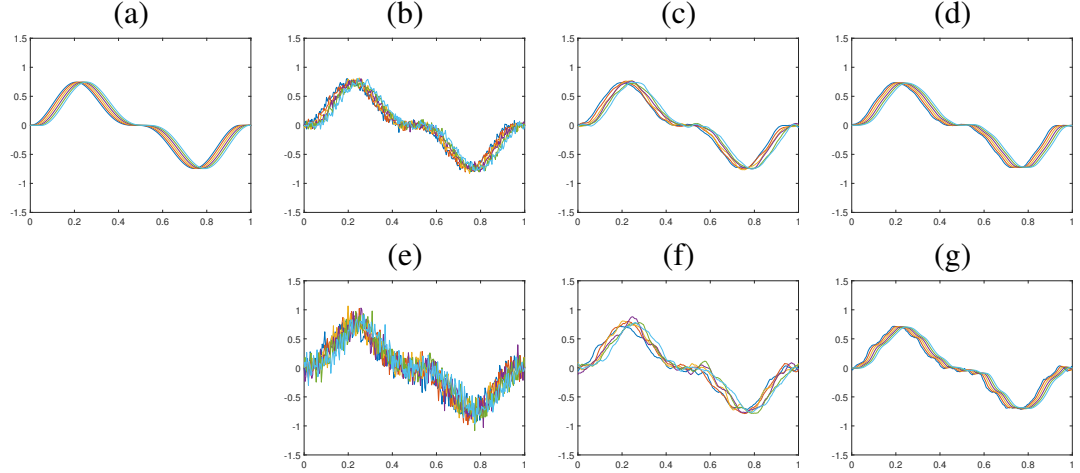


Figure 6.6: Noisy and denoised data of the transport equation (Equation 6.15), as well as simulations of the recovered PDE. (a) The clean data, (b) data with 10% noise, (c) the denoised data $S_x[U]$, (d) simulation of the PDE identified by ST and SC (identical). (e) Data with 30% noise, (f) the denoised data $S_{(x)}[U]$, and (g) simulation of the PDE identified by ST and SC (identical).

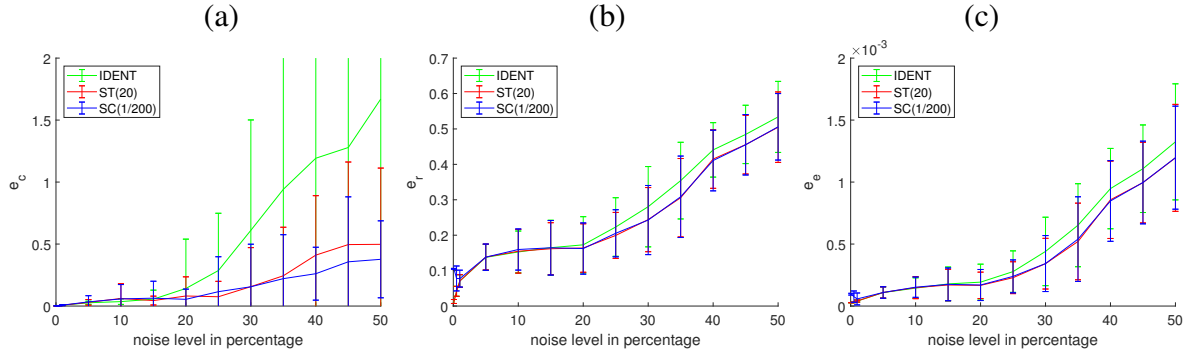


Figure 6.7: The average error e_c , e_r and e_e over 50 experiments for the transport equation (Equation 6.15) with respect to various noise levels. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviation is represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (c) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). The coefficient error e_c by ST and SC is significantly smaller than that of IDENT.

In Figure 6.8, we explore the robustness of SC with respect to the choice of α . We present e_c and e_r versus $1/\alpha$ in (a) and (b) respectively, with 1%, 5%, 10%, 20% noise. Each experiment is repeated 50 times, and the error is averaged. The result shows that SC, in this case, is not sensitive to α , and there are wide range choices of α that give rise to a small error.

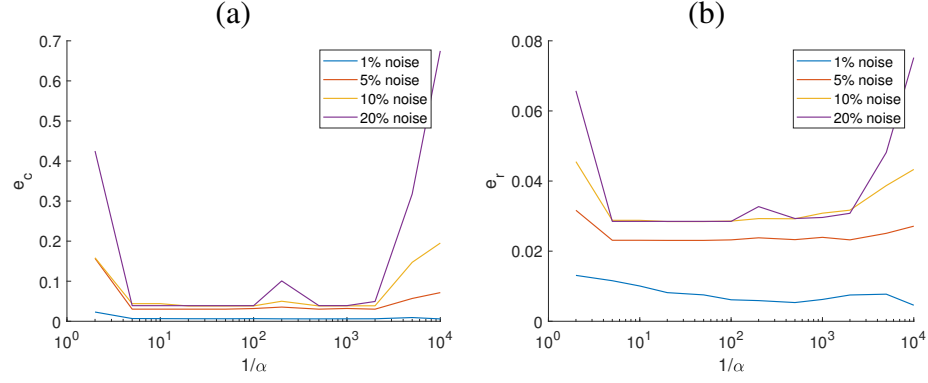


Figure 6.8: Robustness of SC to the choice of α for the recovery of the transport equation (Equation 6.15). (a) and (b) display e_c and e_r versus $1/\alpha$ respectively, with 1% (Blue), 5% (Red), 10% (Orange), 20% (Purple) noise. Each experiment is repeated 50 times, and the errors are averaged. We observe that SC is not sensitive to α , and there is a wide range of values for α that give rise to a small error.

We next test ST and SC on data generated from the transport equation with a discontinuous initial condition. We set the initial condition as

$$u(x, 0) = \begin{cases} \sin^2(2\pi x/(1-T)) \cos(2\pi x/(1-T)), & \text{for } 0 \leq x < (1-T)/3, \\ -\cos^2(2\pi x/(1-T)) + 0.5, & \text{for } (1-T)/3 \leq x < 2(1-T)/3, \\ \sin^2(2\pi x/(1-T)), & \text{for } 2(1-T)/3 \leq x \leq (1-T), \\ 0, & \text{otherwise.} \end{cases} \quad (6.16)$$

The clean data is generated by explicitly solving (Equation 6.15) with $\delta x = \Delta x = 1/256$, $\delta t = \Delta t = 10^{-3}$ and $T = 0.05$. After adding i.i.d. Gaussian noise, we have the noisy data. We show the clean data and the noisy data in Figure 6.9. The identification results are shown in Table 6.3. Even with the existence of discontinuities, ST and SC are stable and can identify the correct PDE with up to 30% noise.

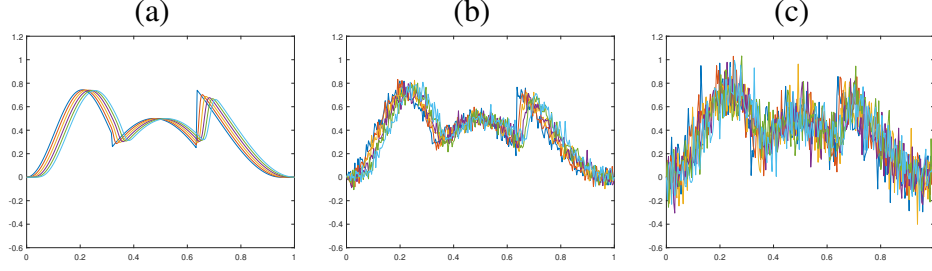


Figure 6.9: Clean and noisy data of the transport equation (Equation 6.15) with the initial condition (Equation 6.16). (a) Clean data. (b) Noisy data with 10% noise. (c) Noisy data with 30% noise.

Table 6.3: Identification of the transport equation (Equation 6.15) with the discontinuous initial condition (Equation 6.16) and different noise levels. In the noise-free case, applying SDD does not introduce strong bias. The identification results (second column) by ST and SC are stable even with 30% noise. Here $w = 20$ for ST, and $\alpha = 1/200$ for SC.

Method	0% noise without SDD	e_c	e_r
ST	$u_t = -1.0091u_x + 9.65 \times 10^{-4}u_{xx}$	1.01×10^{-2}	1.64×10^{-1}
SC	$u_t = -1.0511u_x$	5.11×10^{-2}	4.43×10^{-2}
0% noise with SDD		e_c	e_r
ST, SC	$u_t = -1.0274u_x$	2.74×10^{-2}	1.95×10^{-2}
10% noise		e_c	e_r
ST, SC	$u_t = -0.9913u_x$	8.72×10^{-3}	5.90×10^{-3}
30% noise		e_c	e_r
ST, SC	$u_t = -0.9239u_x$	7.61×10^{-2}	5.36×10^{-2}

6.3.2 Burgers' Equation

In the second example, we test our methods on the Burgers' equation, which is a first-order nonlinear PDE:

$$u_t = -uu_x \quad (6.17)$$

for $0 \leq x \leq 1$ and $0 < t \leq T$. We use the initial condition

$$u(x, 0) = \sin(4\pi x) \cos(\pi x) \quad (6.18)$$

and zero Dirichlet boundary condition. Our data is generated by solving Equation 6.17 with $\delta x = \Delta x = 1/256$, $\delta t = \Delta t = 10^{-3}$ and $T = 0.05$.

Table 6.4: Identification of the Burgers' equation (Equation 6.17) with initial condition (Equation 6.18) and different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 40%. Here $w = 20$ for ST, and $\alpha = 1/500$ for SC.

Method	0% noise without SDD	e_c	e_r
ST	$u_t = -1.0023uu_x - 2.38 \times 10^{-5}u_xu_{xx}$	2.35×10^{-3}	5.07×10^{-3}
SC	$u_t = -0.9960uu_x$	4.01×10^{-3}	2.58×10^{-3}
	0% noise with SDD	e_c	e_r
ST	$u_t = -1.0079uu_x - 0.0001u_xu_{xx}$	7.97×10^{-3}	1.43×10^{-2}
SC	$u_t = -0.9888uu_x$	1.12×10^{-2}	7.20×10^{-3}
	10% noise	e_c	e_r
ST, SC	$u_t = -1.0246uu_x$	2.46×10^{-2}	1.52×10^{-2}
	40% noise	e_c	e_r
ST, SC	$u_t = -0.7366uu_x$	2.63×10^{-1}	1.64×10^{-1}

Table 6.4 shows the results of ST(20) and SC(1/500) with various noise levels. With clean data, ST identifies an additional term, but its coefficient is very small, and the corresponding e_c and e_r are small. SC works very well on clean data. With 10% and 40% noise, both methods identify the same PDE with small e_c and e_r .

Figure 6.10 shows how e_c , e_r and e_e change when the noise level varies. Each experiment is repeated 50 times and the errors are averaged. We test IDENT, ST(20) and SC(1/500). The results in Figure 6.10 show that ST and SC perform better than IDENT.

We then compare SC, ST in this paper with IDENT in [369] and the method proposed in [367]. The method from [367] uses the spectral method to compute the spatial derivatives, which requires periodic boundary conditions. For a fair comparison, we use the initial condition

$$u(x, 0) = \sin(4\pi x) \cos(2\pi x) \quad (6.19)$$

and the periodic boundary condition (in which the boundary values are always 0). Our data is generated by solving Equation 6.17 with $\delta x = \Delta x = 1/256$, $\delta t = \Delta t = 10^{-3}$ and $T = 0.05$. We set $w = 20$ for ST, and $\alpha = 1/500$ for SC. For IDENT, we use SDD to

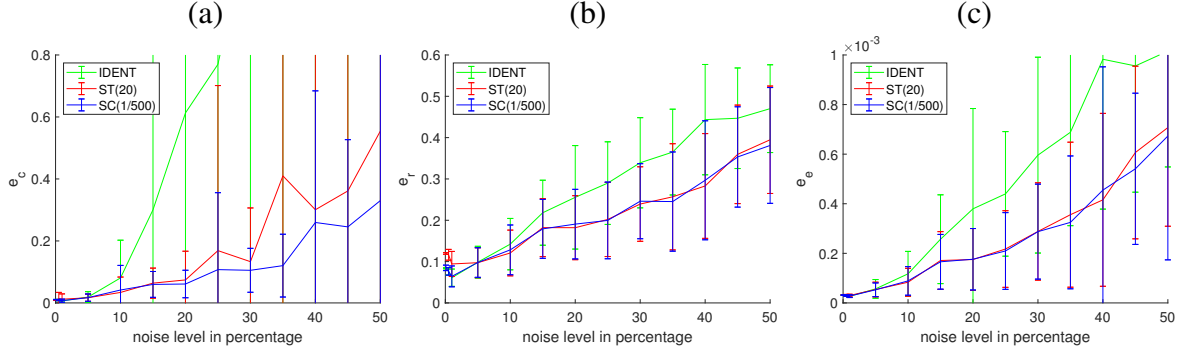


Figure 6.10: The average error e_c , e_r and e_e over 50 experiments for the Burgers' equation (Equation 6.17) with respect to various noise levels, where the initial condition is Equation 6.18. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviations are represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (c) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). The e_c , e_r and e_e of ST and SC are much smaller than those of IDENT.

denoise the data and to compute the partial derivatives, which improves the original IDENT in [369]. For the method in [367], we use the denoising method specified in [367, Example 3.9]. The identification results are shown in Table 6.5. Table 6.5 shows that ST, SC and IDENT are more robust than the method in [367] at various noise levels. The errors given by ST, SC, and IDENT are also smaller.

6.3.3 Burgers' Equation with Diffusion

Our third example is the Burgers' equation with diffusion, which is a second order nonlinear PDE:

$$u_t = -uu_x + 0.1u_{xx} . \quad (6.20)$$

We use the initial condition $u(x, 0) = \sin(3\pi x) \cos(\pi x)$ and zero Dirichlet boundary condition. We first solve Equation 6.20 with $\delta x = 1/256$, $\delta t = 10^{-5}$ and $T = 0.05$. The given data is downsampled from the numerical solution such that $\Delta x = 1/64$ and $\Delta t = 10^{-4}$.

Table 6.6 shows the results of ST(20) and SC(1/10) with various noise levels. With clean data, 1% and 5% noise, both methods identify the PDE with small e_c and e_r .

Figure 6.11 shows how e_c , e_r and e_e change when the noise level varies from 0.1% to

Table 6.5: Comparison of ST, SC with IDENT in [369] and the method in [367] for the identification of the Burgers' equation (Equation 6.17) with the initial condition (Equation 6.19), and various noise levels. In this table, we only include the reconstructed terms with the coefficient magnitudes above 10^{-2} . ST, SC and IDENT are very stable compared to the method in [367]. The coefficient error e_c (Equation 6.13) and the time evolution error e_e (Equation 6.14) are shown. The errors given by ST, SC and IDENT are smaller than the errors given by the method in [367]. Comparison of ST, SC with IDENT in [369] and the method in [367] for the identification of the Burgers' equation (Equation 6.17) with the initial condition (Equation 6.19), and various noise levels. This table only includes the reconstructed terms with the coefficient magnitudes above 10^{-2} . ST, SC, and IDENT are very stable compared to the method in [367]. The coefficient error e_c (Equation 6.13) and the time evolution error e_e (Equation 6.14) are shown. The errors given by ST, SC, and IDENT are smaller than the method's errors in [367].

Method	0% noise	e_c	e_e
[367]	$u_t = -0.95uu_x - 0.01u$	6.55×10^{-2}	1.53×10^{-4}
ST, SC, IDENT	$u_t = -1.0013uu_x$	1.27×10^{-3}	2.62×10^{-5}
	1% noise	e_c	e_e
[367]	$u_t = -0.89uu_x - 0.13u + 0.07u^2$	3.15×10^{-1}	3.42×10^{-4}
ST, SC, IDENT	$u_t = -0.97uu_x$	2.53×10^{-2}	7.38×10^{-5}
	5% noise	e_c	e_e
[367]	$u_t = -0.35uu_x + 0.09u^2 + 0.05u + 0.06$	8.54×10^{-1}	2.00×10^{-3}
ST, SC, IDENT	$u_t = -0.98uu_x$	2.03×10^{-2}	5.95×10^{-5}

Table 6.6: Identification of the Burgers' equation with diffusion (Equation 6.20) with different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 5%. Here $w = 20$ for ST, and $\alpha = 1/10$ for SC.

Method	0% noise without SDD	e_c	e_r
ST, SC	$u_t = -1.0018uu_x + 0.1001u_{xx}$	1.67×10^{-3}	8.14×10^{-4}
	0% noise with SDD	e_c	e_r
ST, SC	$u_t = -0.9994uu_x + 0.1009u_{xx}$	1.36×10^{-3}	7.68×10^{-3}
	1% noise	e_c	e_r
ST, SC	$u_t = -0.9901uu_x + 0.1013u_{xx}$	1.02×10^{-2}	1.19×10^{-2}
	5% noise	e_c	e_r
ST, SC	$u_t = -1.0170uu_x + 0.0976u_{xx}$	1.77×10^{-2}	2.21×10^{-2}

10%. Each experiment is repeated 50 times, and the error is averaged. We test IDENT, ST(20), and SC(1/10). Among the three methods, ST is the best. SC does not perform as well as ST and IDENT when the noise level is large. For high order PDEs, the high order derivatives are heavily contaminated by noise, even with SDD, which affects the accuracy

of cross-validation. While ST and IDENT use time evolution, it is easier to pick correct features. In general, ST performs better than SC for high order PDEs when the given data contain heavy noise.

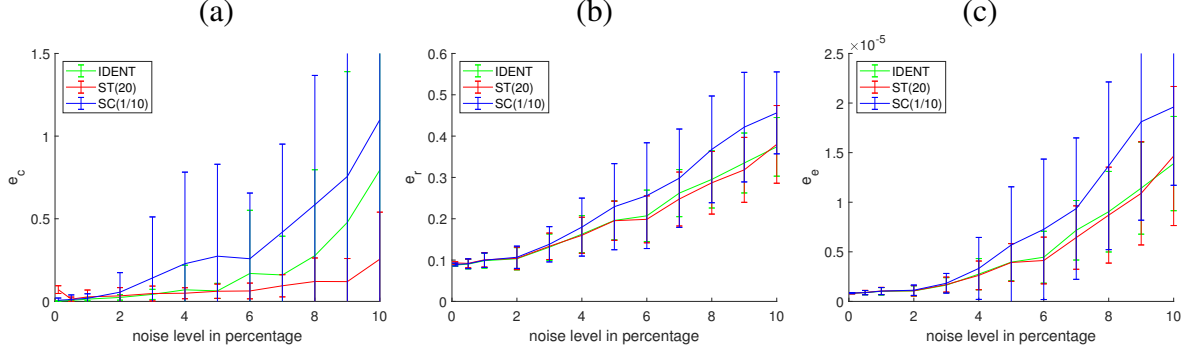


Figure 6.11: The average error e_c , e_r and e_e over 50 experiments of the Burgers' equation with diffusion (Equation 6.20) with respect to various noise levels. (a) The curve represents the average e_c for IDENT [369] (Green), ST (Red) and SC (Blue), and the standard deviations are represented by vertical bars. (b) The average and variation of e_r for IDENT (Green), ST (Red) and SC (Blue). (c) The average and variation of e_e for IDENT (Green), ST (Red) and SC (Blue). Among the three methods, ST gives the best result.

In Figure 6.12, we explore the effect of α in SC on the Burgers' equation with diffusion. Figure 6.12 (a) and (b) show e_c and e_r versus $1/\alpha$ respectively, with 0.5%, 1%, 3%, and 5% noise. When the noise level is low, such as 0.5% and 1%, we have a wide range of good choices of α which gives rise to a smaller error. As the noise level increases, the range of the optimal α becomes narrow.

6.3.4 The KdV Equation

In this example, we test our proposed algorithm to identify the KdV equation

$$u_t + 6uu_x + u_{xxx} = 0, \quad (6.21)$$

on the spatial domain $[-10, 10]$ and the time domain $0 \leq t \leq T$ with $T = 0.4$. We use the initial condition $u(x, 0) = 5\text{sech}^2(1.2x)$. The data is generated with $\delta x = \Delta x = 0.1$, $\delta t = 10^{-5}$. Data are downsampled in the time domain with $\Delta t = 10^{-3}$. Our dictionary contains

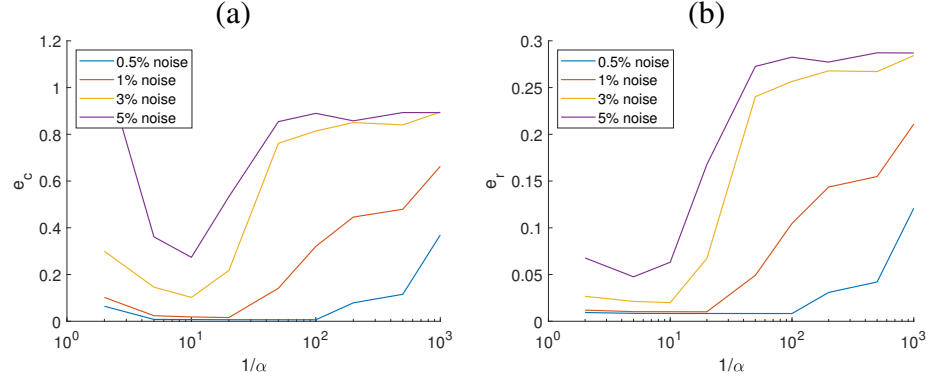


Figure 6.12: Robustness of SC to the choice of α for the recovery of the Burgers' equation with diffusion (Equation 6.20). (a) and (b) display e_c and e_r versus $1/\alpha$ respectively, with 0.5% (Blue), 1% (Red), 3% (Orange), 5% (Purple) noise. Each experiment is repeated 50 times, and the errors are averaged. When the noise level is low, such as 0.5% and 1%, there is a wide range of values for α , which give a small error. As the noise level increases, the range of the optimal α becomes narrow.

$1, u, u_x, u_{xx}$ and u_{xxx} and their pairwise products. There are 15 terms in the dictionary. The identified PDE by ST and SC from clean data is shown in Table 6.7. In this example $w = 20, \widetilde{\Delta t} = 100$ is used in ST and $\alpha = 1/1000$ is used in SC. Our results show that both ST and SC can identify the correct PDE.

Table 6.7: Identification of the KdV equation (Equation 6.21). Both ST and SC can identify the correct PDE.

Method	Identified PDE	e_c	e_r
ST, SC	$u_t = -6.135uu_x - 1.0580u_{xxx}$	2.77×10^{-2}	1.21

6.3.5 A Larger Dictionary

The examples above involve a particular set of a dictionary which consists of the leading terms in the Taylor expansion of the governing equation $f(u, \partial_x u, \partial_x^2 u)$. Our method is general and can be applied if we use other sets of the dictionary.

We next test ST and SC on a larger set of dictionary, including $1, u, u_x, u_{xx}$ and $\sin(2\pi u), \cos(2\pi u)$ and their pairwise products. Since $\sin^2(2\pi u) + \cos^2(2\pi u) = 1$, we exclude the term $\cos^2(2\pi u)$ to guarantee a set of linearly independent features. This dictionary contains

20 features. We consider the following PDE

$$u_t = u - 0.1u_x \sin(2\pi u) \quad (6.22)$$

with initial condition

$$u(x, 0) = 0.8 \sin(3\pi x) \cos(\pi x) \quad (6.23)$$

and zero Dirichlet boundary condition. The data are generated by solving Equation 6.22 with $\delta x = \Delta x = 1/256$, $\delta t = \Delta t = 4 \times 10^{-3}$ and $T = 0.2$. The identified PDEs by ST and SC with various noise levels are shown in Table 6.8. On the clean data without SDD, ST identifies an additional term whose coefficient is very small. The corresponding e_c and e_r are very small. With noise level up to 10%, both ST and SC identify the correct PDE with a small e_c and e_r .

Table 6.8: Identification of Equation 6.22 with different noise levels. The identification results (second column) by ST and SC are good with small e_c and e_r for a noise level up to 5%. Here $w = 20$ for ST, and $\alpha = 1/500$ for SC.

Method	0% noise without SDD	e_c	e_r
ST	$u_t = 0.9994u - 0.0995 \sin(2\pi u)u_x$ $- 2.90 \times 10^{-5} \cos(2\pi u)u_{xx}$	1.01×10^{-3}	1.73×10^{-3}
SC	$u_t = 0.9987u - 0.0992 \sin(2\pi u)u_x$	1.88×10^{-3}	1.13×10^{-3}
	0% noise with SDD	e_c	e_r
ST, SC	$u_t = 0.9903u - 0.0895 \sin(2\pi u)u_x$	1.83×10^{-2}	1.49×10^{-2}
	5% noise	e_c	e_r
ST, SC	$u_t = 0.9909u - 0.0887 \sin(2\pi u)u_x$	1.85×10^{-2}	1.56×10^{-2}
	10% noise	e_c	e_r
ST, SC	$u_t = 1.0646u - 0.1026 \sin(2\pi u)u_x$	6.11×10^{-2}	1.33×10^{-2}

6.3.6 Two Dimensional PDEs

We apply our methods to identify PDEs in a two-dimensional space. The PDEs are solved with $\delta x = \delta y = 0.02$ and $\delta t = 8 \times 10^{-4}$. Data are downsampled from the numerical

solution with $\Delta x = 0.04$ and $\Delta t = 8 \times 10^{-3}$. We fix $w = 10$ for ST and $\alpha = 3/200$ for SC.

The identification of two-dimensional PDEs is more challenging and more sensitive to noise. There are more features in two dimensions, and the directional variation of the data adds complexity to the problem. We will show that both ST and SC are still robust against noise.

We first consider the following PDE:

$$\begin{cases} u_t = 0.02u_{xx} - uu_y \text{ for } (x, y, t) \in [0, 1]^2 \times [0, 0.1], \\ u(x, y, 0) = \sin^2\left(\frac{3\pi x}{0.9}\right) \sin^2\left(\frac{2\pi y}{0.9}\right) \text{ when } (x, y) \in [0, 0.9]^2 \text{ and } 0 \text{ otherwise.} \end{cases}, \quad (6.24)$$

which has different dynamics along the x and y directions. Table 6.9 shows the identification results of ST(10) and SC(3/200) with noise level 0%, 5% and 10%. Both methods identify the same features with small e_c and e_r .

Table 6.9: Identification of the two dimensional PDE (Equation 6.24) with different noise levels. The identification results (second column) by ST and SC have small e_c and e_r for a noise level up to 10%. Here $w = 10$ for ST, and $\alpha = 3/200$ for SC.

Method	0% noise	e_c	e_r
ST, SC	$u_t = 0.0189u_{xx} - 0.9525uu_y$	4.75×10^{-2}	2.48×10^{-2}
	5% noise	e_c	e_r
ST, SC	$u_t = 0.0178u_{xx} - 0.9362uu_y$	8.43×10^{-2}	7.45×10^{-2}
	10% noise	e_c	e_r
ST, SC	$u_t = 0.0134u_{xx} - 0.8674uu_y$	1.33×10^{-1}	1.79×10^{-1}

6.3.7 Identifiability Based on the Given Data

For the PDE identification, especially in high dimensions, the given data U plays an important role. When the initial condition has sufficient variations in each dimension, the correct PDE can be identified. Otherwise, there may be multiple PDEs which generate the same

dynamics. For example, consider the following transport equation without noise:

$$\begin{cases} u_t = -0.5u_x + 0.5u_y, & (x, y) \in [0, 1] \times [0, 1], \quad t \in [0, 0.1] \\ u(x, y, 0) = f(x, y), & (x, y) \in [0, 1] \times [0, 1] \end{cases}, \quad (6.25)$$

where f denotes the initial condition.

Choosing $\delta x = \delta y = 0.02$ and $\delta t = 7 \times 10^{-4}$, taking the downsampling rate in space as 2 and in time as 10, we first choose $f(x, y) = \sin(2\pi x/0.9)^2 \sin(2\pi y/0.9)^2$ for $(x, y) \in [0, 0.9] \times [0, 0.9]$ and 0 otherwise. The identified PDE by SC(1/200) is

$$u_t = -0.5001u_x + 0.4800u_y,$$

where the recovered coefficients are very close to the true coefficients. The same result is identified by using ST(20). Next we choose $f(x, y) = \sin(2\pi x/0.9)^2$ for $(x, y) \in [0, 0.9] \times \mathbb{R}$ and 0 otherwise, then SC(1/200) gives

$$u_t = -0.4992u_x. \quad (6.26)$$

which is also identified by using ST(20). With the specified initial condition, the PDE in Equation 6.25 has the exact solution:

$$u(x, y, t) = \begin{cases} \sin\left(\frac{2\pi(x-0.5t)}{0.9}\right)^2, & x \in [0.5t, 0.9 + 0.5t], \quad (x, y) \in \mathbb{R} \times [0, 1], \quad t \in [0, 0.1] \\ 0, & \text{Otherwise} \end{cases}$$

which also satisfies $u_t = -0.5u_x$. The identified PDE in Equation 6.26 approximates this simpler equation. Since the given data only vary along the x direction, the columns in the feature matrix related to y , e.g., u_y , $u_x u_y$, and u_{yy} , are mostly 0. This explains why our method identifies the PDE in Equation 6.26, instead of Equation 6.25.

In this problem, the original PDE can be identified if the initial condition has sufficient

variations. The identifiability can be defined as follows: Suppose the original PDE is associated with the coefficient vector \mathbf{c}_0 with sparsity S . This PDE is identifiable if there is a unique coefficient vector with sparsity no more than S , such that the evolution of the PDE associated with this coefficient vector, starting from the given initial condition, matches the given data. We believe it is an open question to investigate the theoretical conditions under which the PDE problem is identifiable. Roughly speaking, the PDE problem is identifiable if the PDE solution with a given initial condition gives rise to the feature matrix F , which has a small pairwise coherence, in the sense that any two columns of F have a small correlation. We refer to [369, Theorem 1] for an identifiability condition in Lasso.

6.3.8 Choice of Smoother in SDD

In this paper, we use Moving Least Square (MLS) as the denoising in SDD. To numerically justify this choice among Moving Average (MA) [396], cubic spline interpolation [397], and diffusion smoothing [240], we present the SDD results with these smoothers in Figure 6.13. We first solve the PDE

$$u_t = -0.4uu_x - 0.2uu_y, (x, y) \in [0, 1] \times [0, 1], t \in [0, 0.15], \quad (6.27)$$

with $\Delta t = 0.005$ and $\Delta x = \Delta y = 0.01$, where the initial condition is $u(x, y, 0) = \sin(3\pi x) \sin(5\pi y)$. Then 5% Gaussian noise is added to the numerical solution. Given the noisy data, we perform SDD denoising with different smoothers to obtain various partial derivatives. In MLS, we take the bandwidth $h = 0.04$. For MA, the window size for averaging is fixed to be 3. For Cubic Spline, we use the MATLAB function *csaps* with $p = 0.5$. For the Diffusion denoising, we evolve the noisy surface following the heat equation $u_t = u_{xx} + u_{yy}$ with a time step size $(\Delta x)^2/4$ for 5 iterations. Figure 6.13 shows the SDD results of u, u_x, u_{yy}, uu_x at $t = 0.15$ when different smoothers are used in SDD. All of them recover U (the first row), while MLS preserves the underlying dynamics the best,

i.e., the first and second-order derivatives.

6.4 Support Recovery in Statistics

The discussion in the previous sections leads us to consider a theoretical question: what does sparsity do during the process of PDE identification? This is directly related to the support recovery or variable selection problems of Lasso, which have a long and intensive history in the statistical literature. In the noiseless setting, many researchers [398, 399, 400, 401, 402, 391] established different sufficient conditions for either the deterministic or random predictors for the support recovery problems of linear systems via the ℓ_1 -norm.

Since our work falls into the category of noisy setting, we focus more on reviewing the body of work in the noisy setting. In [403], authors studied the asymptotic behavior of the Lasso-type estimator with fixed dimension K under the general centered i.i.d. noises with variance $\sigma^2 > 0$. Both [404] and [400] independently developed sufficient conditions for the support of Lasso estimator to be contained within true support of the sparse model. Under a more general setting, when the exterior noise is i.i.d. with finite moments, [405] showed that the Irrepresentable Condition [406] is almost necessary and sufficient for Lasso's signed-support recovery for fixed K and s . Furthermore, under the Gaussian noise assumption, they showed that Lasso can still achieve signed-support recovery when K is allowed to grow exponentially faster than n . In a non-asymptotic setting, [407] established the sharp relationship of n , K , and s , required for the exact sign consistency of Lasso, where K and s are allowed to grow as n increases under mutual incoherence condition. Using a similar technique in [407], the paper [408] studied Lasso under Poisson-like model with heteroscedastic noise and show that irrepresentable condition can serve as a necessary and sufficient condition for signed-support recovery in their setting. In the context of graphical model, [409, 410] analyzed the model selection consistency of Gaussian graphical models, and [411] showed the signed-support recovery of

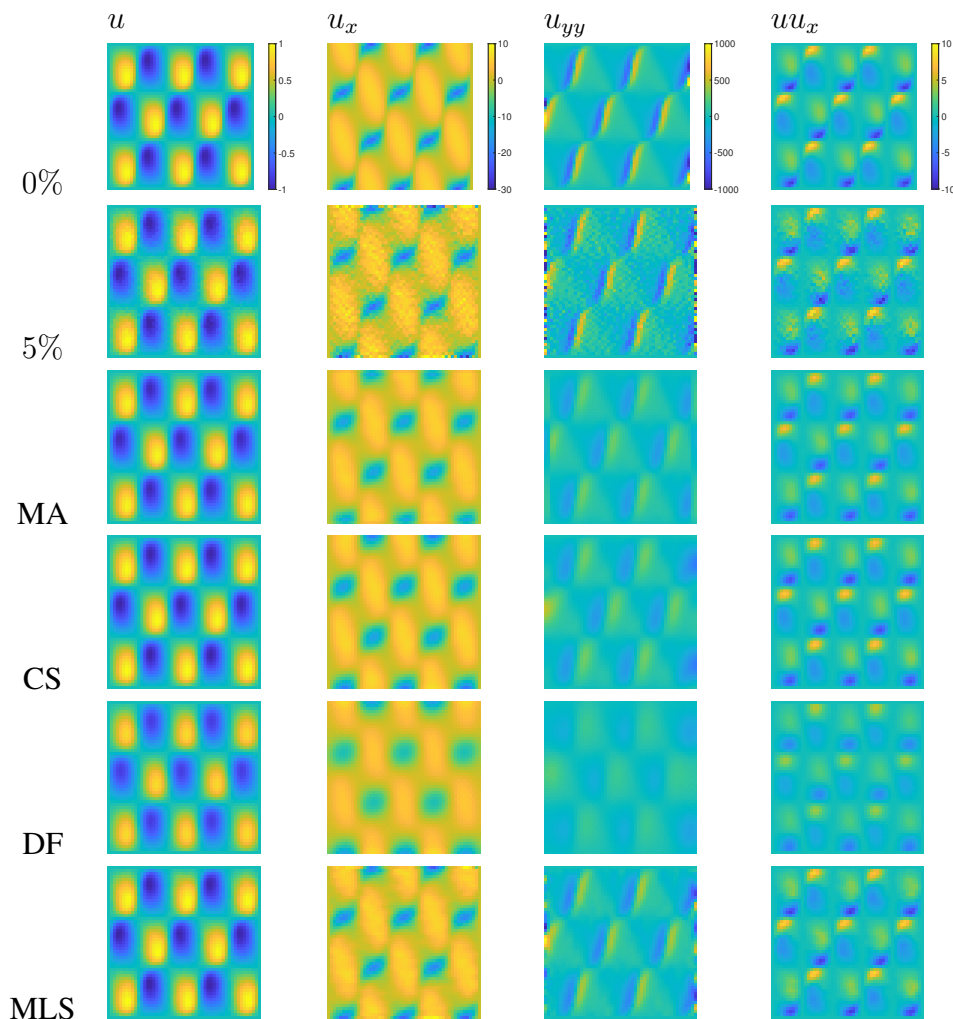


Figure 6.13: SDD results with different smoothers. The first row is the numerical solution of Equation 6.27 at $t = 0.15$ (0% noise) with the initial condition $u_0(x, y) = \sin(3\pi x) \sin(5\pi y)$ and its various partial derivatives. The second row shows the noisy data and its numerical derivatives when 5% Gaussian noise is added to the clean data. The bottom four rows are the SDD results at $t = 0.15$ using MA, cubic spline (CS), diffusion (DF), and MLS in order. While all methods recover U (the first row), the dynamics of the derivatives, especially in the third and fourth rows, are best preserved by MLS.

Ising models. See [412] for a more comprehensive overview on this topic.

6.5 PDE Identification via ℓ_1 -PsLS

6.5.1 Problem Setting

To answer the theoretical question of signed support recovery, we focus on a particular form of PDE identification setting. Take $(x, t) \in [0, X_{\max}) \times [0, T_{\max})$ for some finite constants $0 < X_{\max}, T_{\max} < \infty$. Recall that the underlying mapping \mathcal{F} is a degree 2 polynomial¹ parametrized by a coefficient vector $\beta^* = (\beta_0^*, \beta_1^*, \dots, \beta_{p,q}^*, \dots)$ with real entries, that is,

$$u_t(x, t) = \mathcal{F}(u, \partial_x u, \partial_x^2 u, \dots; \beta^*) := \beta_0^* + \beta_1^* u + \beta_2^* u_x + \beta_3^* u_{xx} + \dots + \beta_{p,q}^* \partial_x^p u \partial_x^q u + \dots \quad (6.28)$$

We call the monomials in the right-hand side of Equation 6.28 as *feature variables*. We set a finite integer upper-bound, $P_{\max} > 0$, for the possible orders of the partial derivatives of u with respect to x in Equation 6.28. Hence, We assume that $\beta^* \in \mathbb{R}^K$, with $K = 1 + 2(P_{\max} + 1) + \binom{P_{\max}+1}{2}$; consequently, constant and any term of the form $\partial_x^p u$ or $\partial_x^p u \partial_x^q u$, for $0 \leq p, q \leq P_{\max}$, are contained in (Equation 6.28). Notice that many entries of β^* can be zero. We denote $\mathcal{S}(\beta^*) := \{0 \leq j \leq K \mid \beta_j^* \neq 0\}$, or simply \mathcal{S} , as the support of the coefficient vector β^* , i.e., the set of indices of the non-zero entries. Additionally, we denote s as the cardinality of the set \mathcal{S} , i.e., $s := |\mathcal{S}(\beta^*)|$.

The given data set $\mathcal{D} = \{(X_i, t_n, U_i^n) \mid i = 0, \dots, M-1; n = 0, \dots, N-1\} \subseteq \Omega \times \mathbb{R}$ consists of $M \times N$ data, where $M, N \in \mathbb{R}$, $M, N \geq 1$. Each $(X_i, t_n) \in \Omega$ represents a space-time sampling grid point, and U_i^n is a representation of $u(X_i, t_n)$ contaminated by additive Gaussian noise:

$$U_i^n = u(X_i, t_n) + \nu_i^n, \quad \nu_i^n \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2),$$

¹It should be noted that our setting can be generalized to higher-degrees of polynomials and functions with multiple spatial dimensions.

whose second moment is uniformly bounded as follows: $\sup_{N,M \in \mathbb{R}} \max_{n,i} E |U_i^n|^2 := \eta^2 < \infty$. Here $\mathcal{N}(0, \sigma^2)$ denotes the centered normal distribution with variance $\sigma^2 > 0$.

6.5.2 Local-Polynomial Regression Estimators for Derivatives

Given data $\{(X_i, t_n, U_i^n)\}$ with $i = 0, 1, \dots, M-1$ and $n = 0, 1, \dots, N-1$, we employ a local quadratic regression to estimate $u_t(X_i, \cdot)$ for each fixed space point X_i and use a Local-Polynomial with degree $p+1$ to estimate $\partial_x^p u(\cdot, t_n)$ at each temporal point t_n , for each degree $p = 0, 1, \dots, P_{\max}$. More specifically, we solve the following optimization problems:

$$\left\{ \widehat{b}_j(X_i, t) \right\}_{j=0,1,2} = \arg \min_{b_j(t) \in \mathbb{R}, 0 \leq j \leq 2} \sum_{n=0}^{N-1} \left(U_i^n - \sum_{j=0}^2 b_j(t)(t_n - t)^j \right)^2 \mathcal{K}_{h_N}(t_n - t),$$

for $i = 0, 1, \dots, M-1$; (6.29)

$$\left\{ \widehat{c}_j^p(x, t_n) \right\}_{j=0,1,\dots,p+1} = \arg \min_{c_j(t) \in \mathbb{R}, 0 \leq j \leq p+1} \sum_{i=0}^{M-1} \left(U_i^n - \sum_{j=0}^{p+1} c_j^p(t)(X_i - x)^j \right)^2 \mathcal{K}_{w_M}(X_i - x),$$

for $n = 0, 1, \dots, N-1$ and $p = 0, 1, \dots, P_{\max}$. (6.30)

and set $\widehat{u}_t(X_i, t) = \widehat{b}_1(X_i, t)$ and $\widehat{\partial_x^p u}(x, t_n) = p! \widehat{c}_p^p(x, t_n)$. Here h_N and $w_{p,M}$ denote the window width parameters, and $\mathcal{K}_w(z) := \mathcal{K}(z/w)/w$ for some kernel function \mathcal{K} with window width $w > 0$. Specific choices of the order of polynomial fit for the functions \widehat{u}_t and $\widehat{\partial_x^p u}$ are to strike the balance between modeling bias and variance. See Subsections 3.1 and 3.3 of Fan and Gijbels [413] for more rigorous treatments on this topic. Also the kernel \mathcal{K} is assumed to be uniformly continuous and absolutely integrable with respect to Lebesgue measure on the real-line; $\mathcal{K}(z) \rightarrow 0$ as $|z| \rightarrow +\infty$; and $\int |z \ln |z||^{1/2} |dK(z)| < +\infty$.

Optimization problems Equation 6.29 and Equation 6.30 have closed-form solutions in the form of weighted least square estimator. See Appendix section C.3. However, for theoretical investigation, we employ the notion of *equivalent kernel* [413, 414] to write the

solutions as follows: for any fixed spatial point $X_i, i = 0, 1, \dots, M - 1$, $\widehat{u}_t(X_i, t)$ can be written as:

$$\widehat{u}_t(X_i, t) = \frac{1}{Nh_N^2} \sum_{n=0}^{N-1} \mathcal{K}_2^* \left(\frac{t_n - t}{h_N} \right) U_i^n \{1 + o_{\mathbb{P}}(1)\}. \quad (6.31)$$

Similarly, for any fixed temporal point $t_n, n = 0, 1, \dots, N - 1$, the estimation for the p -th order partial derivative takes the form:

$$\widehat{\partial_x^p u}(x, t_n) = \frac{p!}{Mw_M^{p+1}} \sum_{i=1}^M \mathcal{K}_p^* \left(\frac{X_i - x}{w_M} \right) U_i^n \{1 + o_{\mathbb{P}}(1)\}. \quad (6.32)$$

Here, $\mathcal{K}_j^*(z) = e_j^T S^{-1}(1, z, \dots, z^p)^T K(z)$ is called an equivalent kernel, where e_j denotes a unit vector with 1 on the j^{th} position; $S = (\int z^{l+s} \mathcal{K}(z) dz)_{0 \leq l, s \leq p}$ is the moment matrix associated with kernel \mathcal{K} ; and $o_{\mathbb{P}}(1)$ denotes a random quantity tending to zero as either N or M tends to infinity. From here, we will omit the dependency on j for the simplicity of notation when using the equivalent kernel.

Remark 6.5.1. *The most important reason for using the Local-Polynomial fitting for the estimation of state variables and their derivatives is its rich literature in asymptotic properties and uniform convergence of the estimator [413, 415, 416, 414]. Specifically, these results allow us to explore the behavior of the tail-probability of the measurement error τ , which is essential for the analysis of the ℓ_1 -PsLS estimator. See Subsection subsection 6.7.2 for more information.*

6.5.3 ℓ_1 -regularized Pseudo Least Square Model

First, we introduce matrix-vector notations for compact expressions of the problem. We let $\mathbf{u}_t \in \mathbb{R}^{NM}$ denote the vectorization of $\{u_t(X_i, t_n)\}_{i=0, \dots, M-1}^{n=0, \dots, N-1}$ in a dictionary order prioritizing the spatial dimension; that is, $\mathbf{u}_t^T = \begin{bmatrix} u_t(X_0, t_0) & u_t(X_1, t_0) & \dots \end{bmatrix}$. Recall the *feature*

matrix, $\mathbf{F} \in \mathbb{R}^{NM \times K}$, as the collection of values of feature variables organized as follows:

$$\mathbf{F} := \begin{bmatrix} 1 & u(X_0, t_0) & \partial_x u(X_0, t_0) & \cdots & \partial_x^p u(X_0, t_0) \partial_x^q u(X_0, t_0) & \cdots \\ 1 & u(X_1, t_0) & \partial_x u(X_1, t_0) & \cdots & \partial_x^p u(X_1, t_0) \partial_x^q u(X_1, t_0) & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots \\ 1 & u(X_{M-1}, t_0) & \partial_x u(X_{M-1}, t_0) & \cdots & \partial_x^p u(X_{M-1}, t_0) \partial_x^q u(X_{M-1}, t_0) & \cdots \\ 1 & u(X_0, t_1) & \partial_x u(X_0, t_1) & \cdots & \partial_x^p u(X_0, t_1) \partial_x^q u(X_0, t_1) & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots \\ 1 & u(X_{M-1}, t_{N-1}) & \partial_x u(X_{M-1}, t_{N-1}) & \cdots & \partial_x^p u(X_{M-1}, t_{N-1}) \partial_x^q u(X_{M-1}, t_{N-1}) & \cdots \end{bmatrix}.$$

With these notations, Equation 6.28 can be written as $\mathbf{u}_t = \mathbf{F}\beta^*$. Note that before estimating the correct signed-support of β^* , \mathbf{u}_t and \mathbf{F} need to be estimated. Conventional regression techniques such as Local-Polynomial regression, smoothing spline, and regression spline, among others, can be used to estimate \mathbf{u}_t and columns of \mathbf{F} . In this paper, we employ the Local-Polynomial approach. We denote $\hat{\mathbf{u}}_t \in \mathbb{R}^{NM}$ and $\hat{\mathbf{F}} \in \mathbb{R}^{NM \times K}$ by replacing the entries of \mathbf{u}_t and \mathbf{F} respectively with the corresponding estimators. (i.e., $(\widehat{u_t})_i^n$, $(\widehat{\partial_x^p u})_i^n$, and $(\widehat{\partial_x^p u})_i^n (\widehat{\partial_x^q u})_i^n$.)

Let $\Delta \mathbf{u}_t = \hat{\mathbf{u}}_t - \mathbf{u}_t$, $\Delta \mathbf{F} = \hat{\mathbf{F}} - \mathbf{F}$ denote the difference between the obtained estimators $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$ via Local-Polynomial regression and their ground-truth counterparts. With these notations, we formally obtain a regression model

$$\hat{\mathbf{u}}_t = \hat{\mathbf{F}}\beta^* + \boldsymbol{\tau}, \quad \text{where } \boldsymbol{\tau} = \Delta \mathbf{F}\beta^* - \Delta \mathbf{u}_t. \quad (6.33)$$

The natural extension for inducing sparsity of the parameter of interest is to add positively weighted ℓ_1 -penalty term $\|\beta\|_1$ to the squared loss $\|\hat{\mathbf{u}}_t - \hat{\mathbf{F}}\beta\|_2^2$, which leads to the following estimator:

$$\hat{\beta}^\lambda = \arg \min_{\beta \in \mathbb{R}^K} \left\{ \frac{1}{2NM} \left\| \hat{\mathbf{u}}_t - \hat{\mathbf{F}}\beta \right\|_2^2 + \lambda_N \|\beta\|_1 \right\}, \quad (6.34)$$

where $\lambda_N > 0$ is a regularization hyper-parameter. Note that we normalize the columns of $\widehat{\mathbf{F}}$ such that $\frac{1}{\sqrt{NM}} \max_{j=1,\dots,K} \|\widehat{\mathbf{F}}_j\|_2 \leq 1$ while solving (Equation 6.34).

Observe that Equation 6.34 is formally identical to Lasso [393] for high-dimensional sparsity recovery. We call (Equation 6.34) as ℓ_1 -**Pseudo Least Square** method instead of Lasso. Similarly with [417], the word *pseudo* comes from the setting of our problem, that is, $\widehat{\beta}^\lambda$ is not a true ℓ_1 -least square estimator, but a minimizer of the ℓ_1 -least square fit with the estimated $\widehat{\mathbf{u}}_t$ and $\widehat{\mathbf{F}}$.

Additionally, the residual vector τ violates conventional assumptions on residuals in linear regression, where they are assumed to be centered and independent among entries. See [405, 407, 403]. If $\widehat{\mathbf{u}}_t$ and $\widehat{\mathbf{F}}$ are unbiased estimators of \mathbf{u}_t and \mathbf{F} , τ is a residual vector with mean zero, but its entries are not independent. However, if $\widehat{\mathbf{u}}_t$ and $\widehat{\mathbf{F}}$ are biased estimators such as Local-Polynomial estimators in our case, τ is not a mean zero random vector. Moreover, the unknown signal β^* makes the distribution of τ completely inaccessible. These complexities make the study of the proposed estimator $\widehat{\beta}^\lambda$ challenging.

6.6 Recovery Theory for ℓ_1 -PsLS based PDE Identification

6.6.1 Signed-Support Recovery

The main goal of this paper is to provide provable guarantees that the proposed ℓ_1 -PsLS method gives asymptotically consistent estimator of β^* in the sense of signed-support recovery. We can formally state this problem with the adoption of $\mathbb{S}_\pm(\beta)$ notation, that is: for any vector $\beta \in \mathbb{R}^K$, we define its extended sign vector, whose each entry is written as:

$$\mathbb{S}_\pm(\beta_i) := \begin{cases} +1 & \text{if } \beta_i > 0 \\ -1 & \text{if } \beta_i < 0 \\ 0 & \text{if } \beta_i = 0, \end{cases}$$

for $i \in \{1, \dots, K\}$. This notation encodes the *signed-support* of the vector β . Denote $\hat{\beta}^\lambda$ as the unique solution of ℓ_1 -PsLS. Then under some regularity conditions on the design matrix \mathbf{F} , we will show,

$$\mathbb{P}[\mathbb{S}_\pm(\hat{\beta}^\lambda) = \mathbb{S}_\pm(\beta^*)] \rightarrow 1 \quad \text{as } N, M \rightarrow +\infty,$$

where N and M denote the grid size of temporal and spatial dimensions, respectively.

6.6.2 Assumptions

We introduce two sufficient conditions frequently assumed in ℓ_1 -regularized regression models for the signed-support recovery of the true signal β^* .

1. **Minimal eigenvalue condition.** There exists some constant $C_{\min} > 0$ such that:

$$\Lambda_{\min}\left(\frac{1}{NM}\mathbf{F}_S^T\mathbf{F}_S\right) \geq C_{\min}. \quad (\text{A1})$$

Here $\Lambda_{\min}(\mathbf{A})$ denotes the minimal eigenvalue of a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, and \mathbf{F}_S is made of columns of \mathbf{F} when the column index is in the support set \mathcal{S} . Note that if this condition is violated, the columns of \mathbf{F}_S would be linearly dependent, and it would be impossible to estimate the true signal β^* even in the “oracle case” when the support set \mathcal{S} is known a priori.

2. **Mutual incoherence condition.** For some *incoherence parameter* $\mu \in (0, 1]$:

$$\left\| (\mathbf{F}_{\mathcal{S}^c}^T \mathbf{F}_S) (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right\|_\infty \leq 1 - \mu. \quad (\text{A2})$$

This condition states that the irrelevant predictors cannot exhibit an overly strong influence on the relevant predictors. More specifically, for each index $j \in \mathcal{S}^c$, the vector $(\mathbf{F}_S^T \mathbf{F}_S)^{-1} \mathbf{F}_S^T \mathbf{F}_j$ is the regression coefficient of \mathbf{F}_j on \mathbf{F}_S , thus, it is a measure of how well the column \mathbf{F}_j aligns with the columns of \mathbf{F}_S . A large μ close to 1

indicates that the columns $\{\mathbf{F}_j, j \in \mathcal{S}^c\}$ are nearly orthogonal to the columns of $\mathbf{F}_\mathcal{S}$, which is desirable for support recovery.

For future reference, we define $\mathcal{Q}^* := (\mathbf{F}_{\mathcal{S}^c}^T \mathbf{F}_\mathcal{S}) (\mathbf{F}_\mathcal{S}^T \mathbf{F}_\mathcal{S})^{-1}$, and name it as *population incoherence matrix*. Also, define its estimated counterpart as $\widehat{\mathcal{Q}}_N := (\widehat{\mathbf{F}}_{\mathcal{S}^c}^T \widehat{\mathbf{F}}_\mathcal{S}) (\widehat{\mathbf{F}}_\mathcal{S}^T \widehat{\mathbf{F}}_\mathcal{S})^{-1}$, and call it *sample incoherence matrix*. Note that the dependence of the support set \mathcal{S} on quantities \mathcal{Q}^* and $\widehat{\mathcal{Q}}_N$ is suppressed for notational simplicity.

6.6.3 Statement of Main Result

Theorem 6.6.1. *Given the observed data set \mathcal{D} whose spatial resolution is related to the temporal resolution via $M = \Theta(N^{\frac{2P_{\max}+5}{7}})$, we take the bandwidths of the kernels in (Equation 6.29) and (Equation 6.30) as $h_N = \Theta(N^{-\frac{1}{7}})$, $w_M = \Theta(M^{-\frac{1}{7}})$, respectively. Under the assumptions (item A1) and (item A2) imposed on the ground-truth feature matrix \mathbf{F} , suppose that the sequence of regularization hyper-parameters $\{\lambda_N\}$ satisfies $\lambda_N = \Omega\left(\frac{C\sqrt{K}\ln N}{\mu N^{2/7-c}}\right)$ for some large enough N , some constant $C > 0$ and $0 < c < \frac{2}{7}$ independent of N . Then, the following properties hold with probability greater than $1 - \mathcal{O}\left(N^{\frac{2P_{\max}+5}{7}} \exp\left(-\frac{1}{6}N^c\right)\right) \rightarrow 1$ as $N \rightarrow \infty$:*

1. *The ℓ_1 -PsLS method (Equation 6.34) has a unique minimizer $\widehat{\beta}^\lambda \in \mathbb{R}^K$ with its support contained within the true support, that is $\mathcal{S}(\widehat{\beta}^\lambda) \subseteq \mathcal{S}(\beta^*)$, and the estimator satisfies the ℓ_∞ bound:*

$$\left\| \widehat{\beta}_\mathcal{S}^\lambda - \beta_\mathcal{S}^* \right\|_\infty \leq K^{3/2} C_{\min} \left(C \frac{\ln N}{N^{2/7-c}} + \lambda_N \right). \quad (6.35)$$

2. *Additionally, if the minimum value of the model parameters supported on \mathcal{S} is greater than the upper-bound of Equation 6.35, that is $\min_{1 \leq i \leq s} |(\beta_\mathcal{S}^*)_i| > K^{3/2} C_{\min} \left(C \frac{\ln N}{N^{2/7-c}} + \lambda_N \right)$, then $\widehat{\beta}^\lambda$ has a correct signed-support. i.e., $\mathbb{S}_\pm(\widehat{\beta}^\lambda) = \mathbb{S}_\pm(\beta^*)$.*

The overall proof sketch of Theorem 6.6.1 is described in the subsection 6.6.4, and relevant technical propositions and lemmas are further provided in section 6.6 and section 6.7.

Here, we give some important remarks about Theorem 6.6.1.

1. The uniqueness claim of $\widehat{\beta}^\lambda$ in (1) seems trivial since the objective function in Equation 6.34 is strictly convex in the regime of K being fixed and $NM \rightarrow \infty$. However, we need to ensure that minimal eigenvalue condition hold over the estimated feature matrix $\widehat{\mathbf{F}}$, given the assumption (item A1) for some $C_{\min} > 0$. We defer this statement as Lemma 6.8.1 in section 6.8 and provide the proof in subsection C.5.1.
2. The first item (1) claims that ℓ_1 -PsLS does not falsely select the arguments in that are not in the support of β^* . Also note that part (2) is a consequence of the sup-norm bound from Equation 6.35: as long as minimum value of $|\beta_i^*|$ over indices $i \in \mathcal{S}$ is not small, ℓ_1 -PsLS is signed-support recovery consistent.
3. The asymptotic orders of M , h_N , and w_M are specifically chosen for simplification. Although there is certain flexibility, the spatial resolution M and the temporal resolution N (as well as h_N and w_M) need to be coordinated well to guarantee the support recovery property. This was expected in practice since we need sufficient sampling frequencies both in temporal and space to estimate the underlying dynamics. Here, the Theorem 6.6.1 present a rigorous justification for a combination of these resolutions which is sufficient for the support recovery.
4. The quantity c is derived from the Tusnady's strong approximation [416] where the error of an empirical distribution is compared with a Brownian bridge in tail probability. See subsection C.4.1. With a larger value of c , the regularization hyper-parameter λ_N needs to remain relatively large, but the convergence is faster. Whereas for a smaller value of c , we can relax the regularization in the cost of a slower probability convergence rate.
5. The threshold of λ_N in the statement of the Theorem shows that when the number of data increases, there is more flexibility in tuning this parameter. If the incoherence

parameter μ is small, or equivalently, the group of correct feature variables and the group of the others are similar, to guarantee that the support of the estimated coefficient vector is contained in the correct one, it suffices to use a large value of λ_N . Such behavior of the threshold is consistent with that described in Theorem 1 of [407].

6. The upper-bound for the ℓ_∞ -norm of the coefficient error in Equation 6.35 consists of two components. The first one concerns the grid resolution determined by N , and the underlying function u as well as the choice of regression kernels encapsulated in the constant C . As N increases to ∞ , this part converges to 0 without explicit dependence on the choice of feature variables selected by ℓ_1 -PsLS. The second component is simple: $K^{3/2}C_{\min}\lambda_N$. When N increases, this part does not vary. This indicates that asymptotically, ℓ_1 -PsLS recovers signed-support of governing PDE, as long as $\min_{1 \leq i \leq s} |(\beta_S^*)_i| > K^{3/2}C_{\min}\lambda_N$.

6.6.4 Proof Strategy of the Main Theorem

The analysis for the proof of Theorem 6.6.1 is naturally divided into two steps as follows: In the first step, we prove a result analogous to that of the Theorem 6.6.1 by imposing incoherence assumption on the estimated feature matrix $\hat{\mathbf{F}}$. Specifically, since $\hat{\mathbf{F}}$ is a random matrix, we assume that for some $\mu \in (0, 1]$, the event, $\{\|\hat{\mathcal{Q}}_N\|_\infty \leq 1 - \mu\}$, holds with some probability at least P_μ , for some $P_\mu \in (0, 1]$. Under this assumption, we prove that the success probability of signed-support recovery of ℓ_1 -PsLS converges to P_μ with an exponential decay rate. This is formally stated as Proposition 6.7.1 in subsection 6.7.1.

In the second step, we show that the success probability P_μ goes to 1, given that the ground-truth matrix \mathbf{F} satisfies assumptions (item A1) and (item A2). This is equivalent to proving that, given the assumptions (item A1) and (item A2) for \mathbf{F} for some $C_{\min} > 0$ and $\mu \in (0, 1]$, the same assumptions hold for the estimated $\hat{\mathbf{F}}$ in probability. We state these results formally in Lemmas 6.8.1 and 6.8.2 in Section section 6.8.

6.7 Analysis Under Sample Incoherence Matrix Assumptions

In this section, we provide a proof overview of Proposition 6.7.1 and the key technical contribution of our paper. All the detailed statements and proofs of the Proposition 6.7.1 and its relevant lemmas are relegated to the Appendix for the conciseness.

6.7.1 Statement of Proposition

We establish the signed-support consistency of ℓ_1 -PsLS estimator when the assumptions are directly imposed on the estimated feature matrix $\hat{\mathbf{F}}$, instead on the ground-truth feature matrix \mathbf{F} . More specifically, we assume that there exist some constants $\mu \in (0, 1]$ and $C_{\min} > 0$, such that the followings hold:

$$\mathbb{P} \left[\left\| \hat{\mathcal{Q}}_N \right\|_{\infty} \leq 1 - \mu \right] \geq P_{\mu} \text{ and } \Lambda_{\min} \left(\frac{1}{NM} \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S \right) \geq C_{\min} \text{ almost surely.} \quad (\text{A3})$$

Here, $P_{\mu} \in [0, 1]$ denotes some probability that $\hat{\mathcal{Q}}_N$ satisfies the incoherence assumption. Equipped with this assumption, we have the following proposition:

Proposition 6.7.1. *Given the observed data set \mathcal{D} , where the spatial resolution is related to the temporal resolution via $M = \Theta(N^{\frac{2P_{\max}+5}{7}})$, we take the bandwidths of the kernels in Equation 6.29 and Equation 6.30 as $h_N = \Theta(N^{-\frac{1}{7}})$, $w_M = \Theta(M^{-\frac{1}{7}})$, respectively. Under the assumptions in Equation A3 imposed on the estimated feature matrix $\hat{\mathbf{F}}$, suppose that the sequence of regularization hyper-parameters $\{\lambda_N\}$ satisfies $\lambda_N = \Omega\left(\frac{C\sqrt{K}\ln N}{\mu N^{2/7-c}}\right)$ for some constant $C > 0$ and $0 < c < \frac{2}{7}$ independent of N . Then, the following properties hold :*

1. *With probability greater than $P_{\mu} - \mathcal{O}\left(N^{\frac{2P_{\max}+5}{7}} \exp\left(-\frac{1}{6}N^c\right)\right) \rightarrow P_{\mu}$ as $N \rightarrow \infty$, the ℓ_1 -PsLS method (Equation 6.34) has a unique minimizer $\hat{\beta}^{\lambda} \in \mathbb{R}^K$ with its support contained within the true support, that is $\mathcal{S}(\hat{\beta}^{\lambda}) \subseteq \mathcal{S}(\beta^*)$.*
2. *With probability greater than $1 - \mathcal{O}\left(N^{\frac{2P_{\max}+5}{7}} \exp\left(-\frac{1}{6}N^c\right)\right) \rightarrow 1$ as $N \rightarrow \infty$, $\hat{\beta}^{\lambda}$*

satisfies the ℓ_∞ bound:

$$\left\| \hat{\beta}_S^\lambda - \beta_S^* \right\|_\infty \leq K^{3/2} C_{\min} \left(C \frac{\ln N}{N^{2/7-c}} + \lambda_N \right). \quad (6.36)$$

3. Additionally, if the minimum value of model parameter supported on \mathcal{S} is greater than the upper-bound of Equation 6.36, that is $\min_{1 \leq i \leq s} |(\beta_S^*)_i| > K^{3/2} C_{\min} \left(C \frac{\ln N}{N^{2/7-c}} + \lambda_N \right)$, then $\hat{\beta}^\lambda$ has a correct signed-support. (i.e., $\mathbb{S}_\pm(\hat{\beta}^\lambda) = \mathbb{S}_\pm(\beta^*)$)

We remark that the first item (1) in Proposition 6.7.1 holds with probability $P_\mu \leq 1$ asymptotically, while the second item (2) holds with probability 1 asymptotically. They are not contradictory, since (1) focuses on the estimation errors on entries within the true support \mathcal{S} , whereas (2) describes the support recovery of the coefficient vector over all indices. Technically speaking, proof of (1) is involved with mutual incoherence condition in Equation A3, whereas (2) is involved with minimum-eigen value condition on $\hat{\mathbf{F}}$ in Equation A3.

6.7.2 Proof Overview of Proposition 6.7.1

Readers can find the proof of Equation 6.36 in subsection C.4.6. Here, we focus on providing the high-level idea on the proof of (1) of Proposition 6.7.1. The most important ingredient for the success of PDW construction is to establish the *strict dual feasibility* of the dual vector \hat{z} , when $\hat{z} \in \partial \|\hat{\beta}^\lambda\|_1$, where $\partial \|\hat{\beta}^\lambda\|_1$ is a sub-differential set of $\|\cdot\|_1$ evaluated at $\hat{\beta}^\lambda$. In other words, we need to ensure that $\|\hat{z}_{S^c}\|_\infty < 1$ with high probability. (See section C.2.) Through Karush–Kuhn–Tucker (KKT) condition of the optimal pair $(\hat{\beta}^\lambda, \hat{z})$ of Equation 6.34 and settings of PDW construction, we can explicitly derive the expression of the dual vector \hat{z} supported on the complement of the support set \mathcal{S} as follows:

$$\hat{z}_{S^c} = \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \hat{z}_S + \underbrace{\frac{1}{\lambda_N M N} \hat{\mathbf{F}}_{S^c}^T \mathbf{\Pi}_{S^\perp} (\Delta \mathbf{u}_t - \Delta \mathbf{F}_S \beta_S^*)}_{:= \tilde{z}_{S^c}}, \quad (6.37)$$

where $\Pi_{\mathcal{S}^\perp}$ is an orthogonal projection operator on the column space of $\widehat{\mathbf{F}}_{\mathcal{S}}$. By the mutual incoherence condition in Equation A3, the first term of the right-hand side in Equation 6.37 is upper-bounded by $1 - \mu$ for some $\mu \in (0, 1]$, with some probability $P_\mu \in [0, 1]$. The remaining task is to control the tail probability of \tilde{Z}_j for $j \in \mathcal{S}^c$: that is to ensure $\mathbb{P}[\max_{j \in \mathcal{S}^c} |\tilde{Z}_j| \geq \mu] \rightarrow 0$ with some exponential decay rate. With the help of Lemma C.4.1 in the Appendix, controlling the probability $\mathbb{P}[\|\tilde{Z}_{\mathcal{S}^c}\|_\infty \geq \mu]$ reduces to controlling $\mathbb{P}[\|\Delta \mathbf{F}_{\mathcal{S}} \boldsymbol{\beta}_{\mathcal{S}}^* - \Delta \mathbf{u}_t\|_\infty \geq \mu \frac{\lambda_N}{\sqrt{K}}]$. Controlling the bound on $\mathbb{P}[\|\boldsymbol{\tau}\|_\infty \geq \varepsilon]$ for some $\varepsilon > 0$ is challenging, since the exact form of the residual distribution $\boldsymbol{\tau}$ is unknown. (Note that $\boldsymbol{\tau} = \Delta \mathbf{F}_{\mathcal{S}} \boldsymbol{\beta}_{\mathcal{S}}^* - \Delta \mathbf{u}_t$ since $\mathbf{u}_t = \mathbf{F} \boldsymbol{\beta}^*$.)

We circumvent this difficulty by using the following inequality: for some thresholds $\varepsilon_N > 0$ and $\varepsilon_M > 0$, both of which go to 0 as N and M tends to ∞ , we have,

$$\begin{aligned} & \mathbb{P}[\|\boldsymbol{\tau}\|_\infty \geq \varepsilon_N + \varepsilon_M] \\ & \leq \mathbb{P}\left[\max_{0 \leq i \leq M-1} \sup_{t \in [0, T_{\max})} |\Delta u_t(X_i, t)| \geq \varepsilon_N\right] + \mathbb{P}\left[\max_{\substack{1 \leq k \leq s \\ 0 \leq n \leq N-1}} \sup_{x \in [0, X_{\max})} |\Delta F_k(x, t_n)| \geq \frac{\varepsilon_M}{s \|\boldsymbol{\beta}^*\|_\infty}\right] \\ & \leq M \cdot \mathbb{P}\left[\sup_{t \in [0, T_{\max})} |\Delta u_t(X_i, t)| \geq \varepsilon_N\right] + sN \cdot \mathbb{P}\left[\sup_{x \in [0, X_{\max})} |\Delta F_k(x, t_n)| \geq \frac{\varepsilon_M}{s \|\boldsymbol{\beta}^*\|_\infty}\right]. \end{aligned}$$

The above inequality naturally leads us to study the uniform convergence of Local-Polynomial estimator to its ground-truth function of interest. Say, for sufficiently large enough grid size of temporal dimension N , for some $\varepsilon_N \geq 0$ that is h_N -dependent threshold and $X_i \in [0, X_{\max})$, we will achieve

$$\mathbb{P}\left[\sup_{t \in [0, T_{\max})} |\widehat{\mathbf{u}}_t(X_i, t) - \mathbf{u}_t(X_i, t)| > \varepsilon_N\right] \rightarrow 0, \quad (6.38)$$

with an exponential decay rate. As for obtaining the exponential decay rate in Equation 6.38, it turns out that thresholds ε_N and ε_M are functions of bandwidth parameters h_N and w_M in Equation 6.31 and Equation 6.32. We choose correct orders of h_N and w_M so that we can ensure that the thresholds ε_N and ε_M go to zero. Then, with the proper

choice on the order of λ_N together with $\mathbb{P}[\|\boldsymbol{\tau}\|_\infty \geq \mu \frac{\lambda_N}{\sqrt{K}}]$, we conclude the proof.

6.7.3 Technical Challenges

Several researchers have tried to achieve uniform convergence of Local-Polynomial or kernel smoothing estimators in almost sure sense. See the works of Masry [418] and Li and Hsing [419].

Here, we provide a high-level idea of the proof of Lemma C.4.2. First, we observe that the higher-order Local-Polynomial smoothing is asymptotically equivalent to higher-order kernel smoothing through equivalent kernel theory [413]. See Equation 6.31 and Equation 6.32 for their equivalences in mathematical form with kernel smoothing estimators. Second, we employ Mack and Silverman's [415] truncation idea on the Local-Polynomial estimator and decompose $\widehat{\mathbf{u}}_t(X_i, t) - \mathbf{u}_t(X_i, t)$ into three parts as follows:

$$\widehat{\mathbf{u}}_t - \mathbf{u}_t = \underbrace{\left(\widehat{\mathbf{u}}_t - \widehat{\mathbf{u}}_t^{B'_N} - \mathbb{E}(\widehat{\mathbf{u}}_t - \widehat{\mathbf{u}}_t^{B'_N}) \right)}_{\text{Asymptotic deviation of truncation error}} + \underbrace{\left(\widehat{\mathbf{u}}_t^{B'_N} - \mathbb{E}\widehat{\mathbf{u}}_t^{B'_N} \right)}_{\text{Asymptotic deviation of truncated estimator}} + \underbrace{\left(\mathbb{E}\widehat{\mathbf{u}}_t - \mathbf{u}_t \right)}_{\text{Asymptotic bias}},$$

where B'_N is some increasing sequence in N , and $\widehat{\mathbf{u}}_t^{B'_N}$ denotes the truncated Local-Polynomial estimator of \mathbf{u}_t . We control the *sup* over $t \in [0, T_{\max})$ on each of these three components. The last component, *Asymptotic bias* of $\widehat{\mathbf{u}}_t$ can be obtained through the classical result from [413, 414]. The exponential decay rate comes from the first two components as follows:

1. *Asymptotic deviation of truncation error* can be decomposed into two parts. The first part, which is $\widehat{\mathbf{u}}_t - \widehat{\mathbf{u}}_t^{B'_N}$, can be easily controlled via chernoff bound of gaussian random variable. by using the definition of truncated estimator $\widehat{\mathbf{u}}_t^{B'_N}$. The second part, which is the expected difference $\mathbb{E}(\widehat{\mathbf{u}}_t - \widehat{\mathbf{u}}_t^{B'_N})$, can be bounded by some deterministic function of B'_N and h_N using the similar arguments in Proposition 1 of [415].

2. *Asymptotic deviation of truncated estimator* is decomposed into two components as well: (1) Brownian Bridge and (2) difference between some two-dimensional empirical process and the Brownian Bridge. (1) can be controlled via uniform convergence of Gaussian Process using the arguments similar to [420], together with simple Markov inequality. (2) can be controlled via Tusnady's strong uniform approximation theory [415, 416], stating that the two-dimensional empirical process can be well approximated by a certain solution path of two-dimensional Brownian-bridge.

Same ideas can be employed for the uniform convergence of $\widehat{\partial_x^p u}$ to $\partial_x^p u$ for $p \geq 0$.

6.8 Uniform Convergence of Sample Incoherence Matrix

In this section, we provide two lemmas that can complete the proof of Theorem 6.6.1. Here, the minimum-eigenvalue and incoherence assumptions are imposed on the ground-truth feature matrix \mathbf{F} , instead on the estimated feature matrix $\widehat{\mathbf{F}}$. See (item A1) and (item A2). That is, there exist $C_{\min} > 0$ and $\mu \in (0, 1]$ such that the followings hold for the unknown support set \mathcal{S} :

$$\Lambda_{\min}\left(\frac{1}{NM}\mathbf{F}_{\mathcal{S}}^T\mathbf{F}_{\mathcal{S}}\right) \geq C_{\min} \quad \text{and} \quad \|\mathcal{Q}^*\|_{\infty} \leq 1 - \mu.$$

Equipped with the above assumptions, we can formally show that success probability of the sample incoherence condition P_{μ} in Equation A3 tends to 1 as $N \rightarrow \infty$. Note that this result is not an immediate consequence of classical random matrix theory (see [421, 422]), since the elements in $\widehat{\mathbf{F}}^T\widehat{\mathbf{F}}$ are highly dependent.

To prove the result, we first need the following lemma asserting that if there exists $C_{\min} > 0$ such that the minimum eigen-value condition holds for $\mathbf{F}_{\mathcal{S}}$, then the sample minimum eigen-value condition holds with probability converging to 1 with an exponential decay rate.

Lemma 6.8.1. *Suppose that the assumption (item A1) holds with some constant $C_{\min} > 0$ and $0 < c < \frac{2}{7}$, then with probability at least $1 - \mathcal{O}(N \exp(-\frac{1}{6}N^c))$, we have,*

$$\Lambda_{\min}\left(\frac{1}{NM}\widehat{\mathbf{F}}_S^T\widehat{\mathbf{F}}_S\right) \geq C_{\min} .$$

With the help of Lemma 6.8.1, we can show that the sample incoherence condition holds with high probability, given that there exists $\mu \in (0, 1]$ for the ground-truth version of (item A2).

Lemma 6.8.2. *Suppose that the assumption (item A2) holds with some constant $\mu \in (0, 1]$ and $0 < c < \frac{2}{7}$, then with probability at least $1 - \mathcal{O}(N \exp(-\frac{1}{6}N^c))$, we have,*

$$\left\|\widehat{\mathcal{Q}}_N\right\|_{\infty} \leq 1 - \mu .$$

Verification of Lemma 6.8.2 automatically leads to the complete proof of Theorem 6.6.1, together with Proposition 6.7.1. Therefore, as long as the two assumptions (item A1) and (item A2) hold for \mathbf{F} , with sufficiently fine-grained grid points over the function $u(X, t)$, ℓ_1 -PsLS can always find the correct signed-support of the given PDE model, with the minimum absolute value of β_S^* not too close to zero.

Remark 6.8.1. (Technical Difficulties of Lemma 6.8.1 and 6.8.2.) *The proof procedure is involved with controlling the tail probability of difference between inner-product of two arbitrary columns of $\widehat{\mathbf{F}}$ and inner-product of the two corresponding columns of ground-truth \mathbf{F} . This problem is challenging even if the exact distribution of any entries of $\widehat{\mathbf{F}}$ is known, since the distribution of $\sum_{k=1}^{NM} \widehat{F}_{ki} \widehat{F}_{kj}$ needs to be derived. We circumvent this problem by taking the advantage of the uniform convergence result of $\widehat{\partial_x^p u}$ for any $p \geq 0$.*

6.9 Numerical Experiments

6.9.1 Experimental Setting

In this subsection, we provide detailed descriptions on (1) two popular PDE models that we are going to work on throughout the experiments, and on (2) how to generate the data from respective models, and (3) how to design the regression problem for the experiments to be presented.

We consider the following viscous Burgers' equation:

$$u_t = -uu_x + \nu u_{xx} , \quad 0 < x < 1, 0 < t < 0.1 \quad (6.39)$$

$$u(x, 0) = \sin^2(2\pi x) + \cos^3(3\pi x) , \quad 0 \leq x \leq 1 , \quad u(0, t) = u(1, t) , \quad 0 \leq t \leq 0.1.$$

and the Korteweg–de Vries equation whose dimensionless form is given as

$$u_t + u_{xxx} + 6uu_x = 0 . \quad (6.40)$$

with the initial condition:

$$u(x, 0) = 3.5 \sin^3(4\pi x) + 1.5 \exp \left(-\sin(2\pi x)(1 - x) \right) ,$$

$$0 \leq x \leq 1 , \quad u(0, t) = u(1, t) , \quad 0 \leq t \leq 0.1.$$

For N -size sampling in the temporal dimension, by Theorem Theorem 6.6.1, we take $M = \lfloor N^{(2 \times P_{\max} + 5)/7} \rfloor$ sample size in the space dimension. We numerically solve Viscous Burgers' equation (Equation 6.39) by the Lax-Wendroff scheme on a grid with interval width $\delta t = 0.1/(100N)$ in temporal and $\delta x = 1/M$ in space, then we downsampled the data in the temporal dimension by a factor of 100; thus the resulted clean data is distributed over a grid with N nodes in temporal and M nodes in space. Lastly, we added i.i.d. Gaussian noise with standard deviation $\sigma = 0.25$ to the data. i.e., $\nu_i^n \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.25^2)$. As

for solving the KdV equation (Equation 6.40), the same approaches with Viscous Burger's equation are applied, with i.i.d. Gaussian noises with standard deviation $\sigma = 0.025$.

Constructions of Regression Problems

We employ the Local-Polynomial smoothing for estimating $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$ as described in Subsection 6.5.2. Regarding a choice of kernel for constructing $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$, we use the Epanechnikov kernel defined by:

$$\mathcal{K}(z) = \frac{3}{4}(1 - z^2)_+, \quad z \in \mathbb{R},$$

where $(\cdot)_+ := \max(0, \cdot)$. Bandwidth parameters h_N and w_M in Equation 6.29 and Equation 6.30 are chosen in the order of $h_N = \Theta(N^{-\frac{1}{7}})$ and $w_M = \Theta(M^{-\frac{1}{7}})$, respectively. As displayed in Table 6.10, for the experiments presented in this Section, we choose specific constant factors in the order expressions of h_N and w_M for Viscous Burgers equation and KdV equation. It is also worth noting that we do not use Equation 6.31 and Equation 6.32 as solutions of the optimization problems (Equation 6.29) and (Equation 6.30) for the experiments, since the expressions in Equation 6.31 and Equation 6.32 are derived in asymptotic settings. For the reader's convenience, We provide the closed form solutions of Equation 6.31 and Equation 6.32 in section C.3.

For Viscous Burgers' equation, on the set of noisy data, Local-Polynomial fitting with $P_{\max} = 2$ is applied to construct $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$. Specifically, our goal is to identify the fifth and the sixth coefficients, β_5 and β_6 , of a following linear measurement via our proposed ℓ_1 -PsLS model (Equation 6.34):

$$\hat{\mathbf{u}}_t = \beta_0 + \beta_1 \hat{\mathbf{u}} + \beta_2 \hat{\mathbf{u}}^2 + \beta_3 \hat{\mathbf{u}}_x + \beta_4 \hat{\mathbf{u}}_x^2 + \beta_5 \hat{\mathbf{u}} \hat{\mathbf{u}}_x + \beta_6 \hat{\mathbf{u}}_{xx} + \beta_7 \hat{\mathbf{u}}_{xx}^2 + \beta_8 \hat{\mathbf{u}}_x \hat{\mathbf{u}}_{xx} + \beta_9 \hat{\mathbf{u}} \hat{\mathbf{u}}_{xx}.$$

For KdV equation, after generating the data-points, $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$ are fitted through Local-Polynomial with $P_{\max} = 3$. We want ℓ_1 -PsLS to select β_5 and β_{10} as non-zero coefficients

Table 6.10: Specific choices of the constants in the order of $h_N = \Theta(N^{-\frac{1}{7}})$ and $w_M = \Theta(M^{-\frac{1}{7}})$ for the experiments on Viscous Burgers equation and KdV equation are presented.

	w_M	h_N
Viscous Burgers	$0.75M^{-\frac{1}{7}}$	$0.25N^{-\frac{1}{7}}$
KdV	$0.1M^{-\frac{1}{7}}$	$0.01N^{-\frac{1}{7}}$

in a following linear measurement:

$$\begin{aligned}\hat{\mathbf{u}}_t = & \beta_0 + \beta_1 \hat{\mathbf{u}} + \beta_2 \hat{\mathbf{u}}^2 + \beta_3 \hat{\mathbf{u}}_x + \beta_4 \hat{\mathbf{u}}_x^2 + \beta_5 \hat{\mathbf{u}} \hat{\mathbf{u}}_x + \beta_6 \hat{\mathbf{u}}_{xx} + \beta_7 \hat{\mathbf{u}}_{xx}^2 + \beta_8 \hat{\mathbf{u}}_x \hat{\mathbf{u}}_{xx} + \beta_9 \hat{\mathbf{u}} \hat{\mathbf{u}}_{xx} \\ & + \beta_{10} \hat{\mathbf{u}}_{xxx} + \beta_{11} \hat{\mathbf{u}}_{xxx}^2 + \beta_{12} \hat{\mathbf{u}}_x \hat{\mathbf{u}}_{xxx} + \beta_{13} \hat{\mathbf{u}}_{xx} \hat{\mathbf{u}}_{xxx} + \beta_{14} \hat{\mathbf{u}} \hat{\mathbf{u}}_{xxx}.\end{aligned}$$

6.9.2 Numerical Verifications of Main Statements

In this subsection, we design an experiment to numerically verify following two main statements of this paper.

1. *Under the assumptions (item A1) and (item A2), and with large enough data points, there exist some $\lambda \geq 0$ such that ℓ_1 -PsLS model (Equation 6.34) recovers a signed-support ($\mathbb{S}_{\pm}(\hat{\beta}) = \mathbb{S}_{\pm}(\beta^*)$) of an unique PDE that admits the underlying function as a solution in probability.*
2. *Given the assumptions (item A2) for some $\mu \in (0, 1]$, sampled incoherence parameter μ' converges to ground-truth incoherence parameter μ in probability with large enough data points.*

The experiment is conducted over two PDE models, Viscous Burgers' equation and KdV equation introduced in subsection 6.9.1. We generate the data by setting $\nu = 0.03$ in Equation 6.39. In Figure 6.14, the probability of signed-support recovery $\mathbb{P}[\mathbb{S}_{\pm}(\hat{\beta}) = \mathbb{S}_{\pm}(\beta^*)]$ versus the grid size of temporal dimension N , and $\|\hat{z}_{\mathcal{S}^c}\|_{\infty}$ versus N are recorded on the same plot for respective models. Each point on each curve, which represents $\mathbb{P}[\mathbb{S}_{\pm}(\hat{\beta}) = \mathbb{S}_{\pm}(\beta^*)]$, in (a) and (b) corresponds to the average over 100 trials. For each iteration, the hyper-parameter λ_N is chosen in an optimal way: we used the value yielding

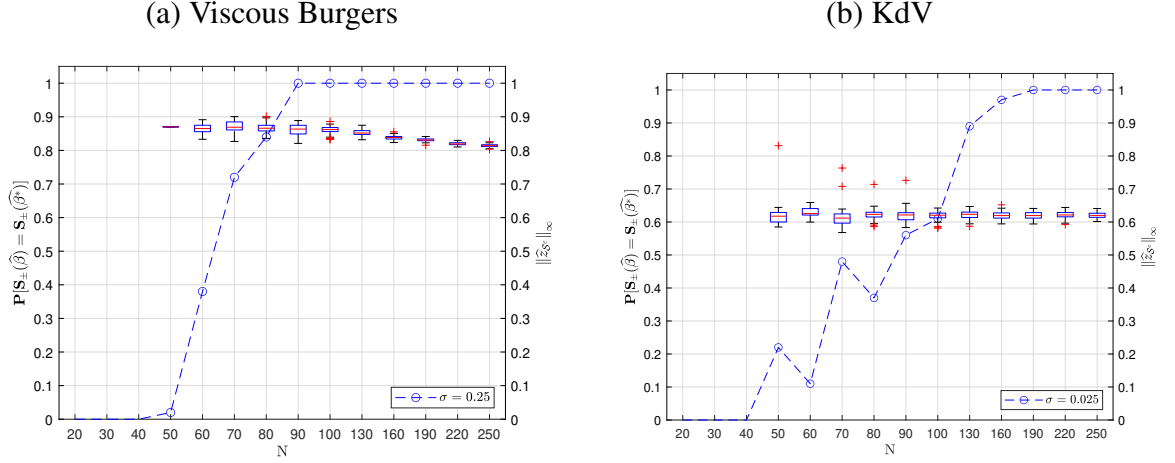


Figure 6.14: Probability of signed-support recovery $\mathbb{P}[\mathbb{S}_{\pm}(\hat{\beta}) = \mathbb{S}_{\pm}(\beta^*)]$ versus the grid size of temporal dimension N , and $\|\hat{z}_{\mathcal{S}^c}\|_{\infty}$ versus N are recorded on the same plot for Viscous Burger's equation in panel (a) and for KdV equation in panel (b), respectively.

the correct number of nonzero coefficient. With the chosen λ_N , $\hat{z}_{\mathcal{S}^c}$ is calculated as given in Equation 6.37. Note that Equation 6.37 can be calculated only when the ℓ_1 -PsLS finds λ_N that gives the minimizer of Equation 6.34 $\hat{\beta}^{\lambda}$ such that $\hat{\beta}_{\mathcal{S}^c}^{\lambda} = 0$ and $\mathcal{S}(\hat{\beta}^{\lambda}) \subseteq \mathcal{S}(\beta^*)$. For this reason, boxplots of $\|\hat{z}_{\mathcal{S}^c}\|_{\infty}$ in (a) and (b) are drawn from the point when ℓ_1 -PsLS starts to find such λ_N . For both models, $\mathbb{P}[\mathbb{S}_{\pm}(\hat{\beta}) = \mathbb{S}_{\pm}(\beta^*)]$ goes to 1, as we observe more data points on finer grid. Furthermore, it is worth noting that the *strict dual feasibility* condition (i.e., $\|\hat{z}_{\mathcal{S}^c}\|_{\infty} < 1$) holds for both cases. In Figure 6.15, boxplots of $\|\hat{Q}_N\|_{\infty}$ versus N are displayed for Viscous Burgers' equation and KdV equation respectively. A dotted horizontal line in each panel represents $1 - \mu$ calculated from the ground-truth feature matrix \mathbf{F} . Notice that as the number of observed data gets larger, the sampled incoherence parameter goes below the dotted lines for both models.

6.9.3 Impact of β_{\min}^* in Signed-Support Recovery of ℓ_1 -PsLS

Theorem 6.6.1 states that as long as $\beta_{\min}^* := \min_{i \in \mathcal{S}} |\beta_i^*|$ is beyond certain threshold, ℓ_1 -PsLS is signed-support recovery consistent. In this subsection, we design an experiment to numerically confirm this claim. The experiment is performed over Viscous Burgers'

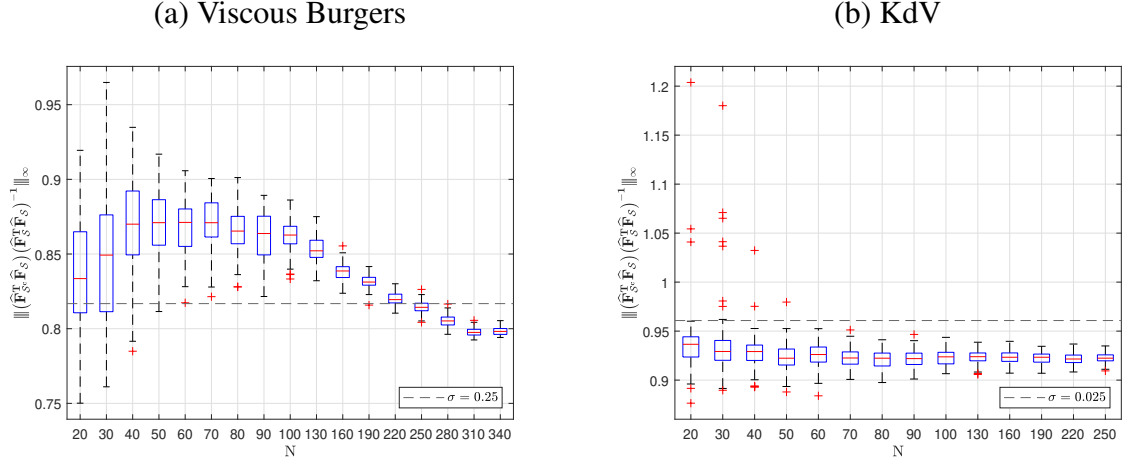


Figure 6.15: Boxplots of $\|\hat{\mathbf{Q}}_N\|_\infty$ versus N are displayed for Viscous Burgers' equation in panel (a) and KdV equation in panel (b), respectively.

equation by varying the coefficient ν in Equation 6.39 : we set $\nu = 0.03, 0.02, 0.01, 0.005$. The Figure Figure 6.16 (a) displays the curves representing $\mathbb{P}[\mathbb{S}_\pm(\hat{\beta}) = \mathbb{S}_\pm(\beta^*)]$ versus N for each of the four cases. Each point on each curve represents the average over 100 trials. The Figure 6.16 (b) exhibits the range of λ_N for which ℓ_1 -PsLS finds the support of $\hat{\beta}^\lambda$ that is contained within the true support, when ν is set as 0.005. More specifically, boxplots in (b) record the range of λ_N that picks $\hat{\mathbf{u}}_{xx}$ as the selected argument. In (a), we can check that, as the magnitude of $\min_{i \in \mathcal{S}} |\beta_i^*|$ decreases from 0.03 to 0.01, ℓ_1 -PsLS requires more data-points for the signed-support recovery, and when $\min_{i \in \mathcal{S}} |\beta_i^*|$ drops to 0.005, ℓ_1 -PsLS fails to recover the governing PDE. On the other hand, (b) says that there exists a range of λ_N for which ℓ_1 -PsLS can still recover a subset of β^* , while the perfect signed-support recovery is difficult.

6.10 Summary

In the first half of this chapter, we introduced two robust methods for PDE identification when noisy data are given. First, we described a Successively Denoised Differentiation (SDD) procedure to stabilize numerical differentiation, which significantly improves the accuracy in the computation of the feature matrix from noisy data. We then discussed

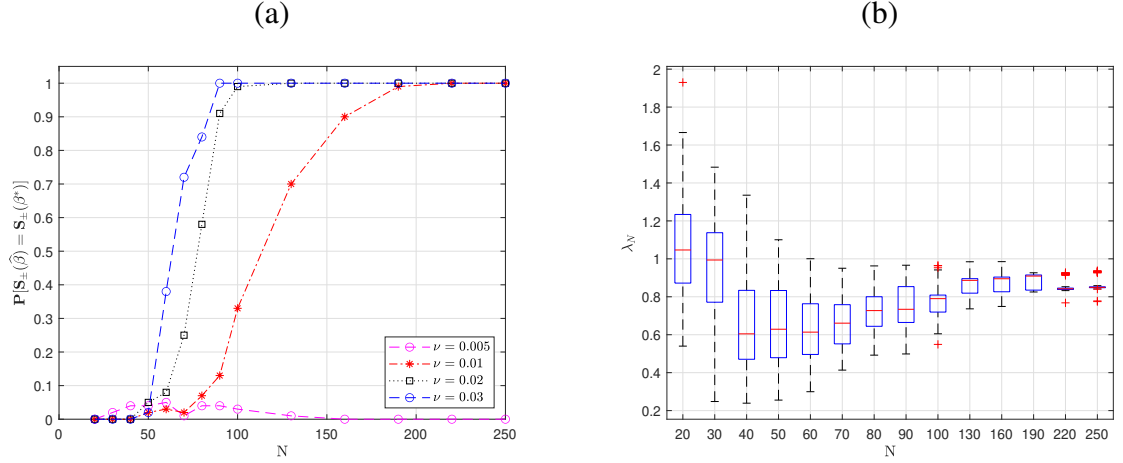


Figure 6.16: Left panel (a) displays the curves representing $\mathbb{P}[S_{\pm}(\hat{\beta}) = S_{\pm}(\beta^*)]$ versus N , when $\nu = 0.03, 0.02, 0.01, 0.005$. Right panel (b) exhibits the range of λ_N for which ℓ_1 -PsLS gives the solution $\hat{\beta}^\lambda$ such that $S(\hat{\beta}^\lambda) \subseteq S(\beta^*)$ with respect to N , when ν is set as 0.005.

two new robust PDE identification algorithms called ST and SC. These algorithms utilize the Subspace Pursuit (SP) greedy algorithm to select a candidate set and then refine the results by time evolution or cross-validation. We presented various numerical experiments to demonstrate the effectiveness of both methods. SC is more computationally efficient, while ST performs better for PDEs with high order derivatives. We also provided an error analysis of ST and SC in the context of PDE identification, which unifies many related methods in the literature.

In the second half, we provided a formal theoretical analysis on the PDE identification via ℓ_1 -regularized Pseudo Least Square method from the statistical point of view. In this article, we assume that the differential equation governing the dynamic system can be represented as a linear combination of various linear and nonlinear differential terms. We employ the Local-Polynomial fitting and apply the ℓ_1 penalty for model selection. A signed-support recovery of ℓ_1 -PsLS method with an exponential convergence rate is obtained under the classical mutual incoherence condition on the feature matrix \mathbf{F} . We divide the cases into two for the proof of the Theorem 6.6.1. Firstly, a signed-support recovery of ℓ_1 -PsLS method is shown with mutual incoherence assumption being imposed on the

estimated feature matrix $\hat{\mathbf{F}}$. Then, we show $\hat{\mathcal{Q}}_N$ gets close to \mathcal{Q}^* under $\|\cdot\|_\infty$ ensuring the statement of the Theorem 6.6.1. We run numerical experiments on two popular PDE models, and the results from the experiments corroborate our theoretical predictions.

CHAPTER 7

DEEP SPATIAL-TEMPORAL SYNTHESIZER FOR DYNAMIC PET RECONSTRUCTION

Positron emission tomography (PET) is a non-invasive imaging technique that measures various biochemical and physiological activities via capturing gamma rays generated by annihilation of positrons emitted from the nucleus of radioisotope [423, 424, 425]. Having no pharmacological actions, these isotopes are injected into the bloodstream as *tracers* detected by the gamma scanner. The variation of radiotracer concentration provides an enormous amount of physiological information including oxygen consumption rate [426] and tumor growth rate [427]. Diagnosis and treatment in cancers [428], heart diseases [429], brain disorders [430] and many other fields have found PET valuable. In terms of the acquired materials, there are mainly two types of PET: *static PET* [424] and *dynamic PET* (dPET) [425]. Contrary to static PET, which renders single images, dPET records the kinetic data of the tracer in the body and produces image sequences. dPET is widely applied in respiratory motion monitoring [431], myocardial blood flow examination [432], and neurotransmitter response [433].

Reconstruction of the tracer’s distribution, or pixel-wise *time activity curves* (TACs), from dPET projection data is a challenging inverse problem, as the decay of biochemical is very fast compared to that of data acquisition. Typically, the reconstruction methods involve minimizing a regularized functional [434, 435, 436, 437, 438, 439]. Depending on the noise model, both L_2 -norm [440] and Kullback–Leibler divergence [438, 437] are used for data fidelity. To amend the problem’s ill-posedness, many regularization terms were explored, such as total variation (TV) norm [437, 441, 442], nonlocal TV [443], and in-class discrepancy [437]. Considering spatial and temporal relationship between the dynamic images, non-negative matrix factorization (NMF) [442, 437, 444] is a popular strategy where

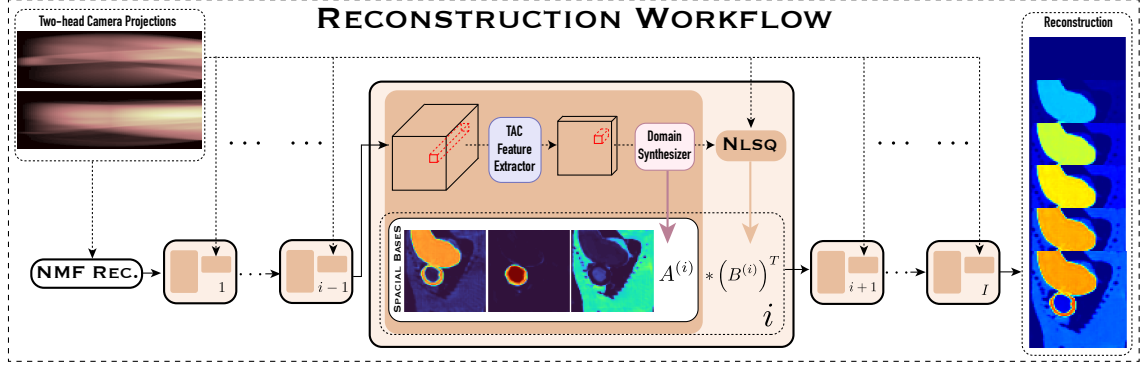


Figure 7.1: Workflow of the proposed STIS model.

the reconstructed image sequence is assumed to have a low-rank decomposition into spatial and temporal bases. It often requires multiple regularizers, thus many manually-tuned parameters, to produce reasonable results due to instability caused by the sparse sampling and noise [442, 437]. The resulted models are often non-convex and challenging to solve with guaranteed convergence.

Recently, deep learning shows great success in feature extraction, which inspired researchers to devise many architectures that combine physical model to improve reconstruction quality in medical images [445, 444]. For example, based on the NMF model, a unsupervised dPET reconstruction paradigm supplied with dense sampling [444] was proposed to generate regularized spatial bases using parallel U-Nets without CT/MRI references. However, there is limited work on dPET reconstruction from sparse sampling, e.g., each frame is only projected by a two-head camera [442].

In this chapter, we take the deep learning perspective and introduce a particularly interesting neural architecture proposed in [446]. It allows to extract spatial and temporal features of the projection data from PET image sequences in an effective way.

7.1 Workflow Overview

Starting from a preliminary NMF reconstruction, the temporal information is encoded by a NN, called *TAC feature extractor*, for identifying the pointwise kinetic features from a

rough reconstruction. A spatial encoder-decoder, called *domain synthesizer*, follows to locally integrate these features and identify homogeneous regions as the spatial bases. The proposed method is thus named as **S**pacial and **T**emporal **I**nformation **S**ynthesis (STIS). The reconstruction of dPET can be obtained by combining the spatial bases and temporal bases via a non-negative least-square fitting. The updated temporal bases reveal better underlying TAC dynamics, which facilitate TAC feature extraction; hence we augment the model by alternatively iterating spatial and temporal bases updates. Figure 7.1 shows the proposed model's workflow, where I is the maximal number of iteration and the block for the i -th iteration is expanded for details.

This method enjoys the interpretability based on NMF and the flexibility of DNNs for both temporal and spatial feature synthesis. Moreover, the proposed network is easy to train thanks to the reduced number of parameters resulted from temporal-spatial separation. The numerical results on sparsely sampled projection validate the proposed method's advantages over other state-of-art methods applied for dPET reconstruction.

7.2 Nonnegative Matrix Factorization for dPET

7.2.1 Imaging Model for Sparsely Sampled dPET

Let $u(x, t)$ denote the tracer distribution at location $x \in \Omega$ and at time $0 \leq t \leq T_{\max}$. The TAC at x refers to the curve $\{u(x, \cdot) : 0 \leq t \leq T_{\max}\}$. Here $\Omega = [0, H] \times [0, W]$ is a rectangular image domain with H, W both positive integers, and $T_{\max} > 0$ is the maximum observing time. dPET projection of u by a single camera initially pointing at angle θ_0 and rotating with angular speed $\Delta\theta$ is modeled by the time-dependent attenuated Radon transform [441]

$$\mathcal{R}u(\theta_0 + t\Delta\theta, s, t) = \int_{L(\theta_0 + t\Delta\theta, s)} u(x, t) \exp(-\mu H(x)) dx . \quad (7.1)$$

Here $s \in \mathbb{R}$ is the collimator coordinate; the projection line $L(\theta, s) = \{t\theta + s\theta^\perp : t \in \mathbb{R}\}$ has angle $\theta \in [0, \pi)$; $H(x)$ is the distance from x to the collimator; and $\mu \geq 0$ is a constant grade of attenuation of gamma rays. We note that the projection angle in Equation 7.1 varies with time, and the sparsely sampled dPET data refers to a sampling paradigm where only a limited number of cameras is available.

7.2.2 NMF Reconstruction

As homogeneous tissues present approximately uniform radiotracer concentration throughout the observation [447], the interplay between temporal and spatial characteristics of dPET data implies an effective dimension reduction.

Upon discretization, for positive integers N, M , we assume that the original activity map consists of N frames: $\{U_n\}_{n=1}^N$, each of which is a 2D image with M pixels. Let $U = \begin{bmatrix} \text{vec}(U_1) & \text{vec}(U_2) & \cdots & \text{vec}(U_N) \end{bmatrix}$, where $\text{vec}(U_n) \in \mathbb{R}^M$ denotes the vectorization of the image U_n obtained by stacking its columns on top of one another. We assume that U admits a non-negative factorization $U = AB^T$ where $A \in \mathbb{R}^{M \times K}$ whose columns are the spatial bases and $B \in \mathbb{R}^{N \times K}$ whose columns are the temporal bases, are matrices with non-negative entries. Here $K \ll N$ and M , thus it offers a low-rank representation of U . Suppose there are $C \geq 1$ cameras initially posed at angles θ_c , $c = 1, 2, \dots, C$, rotating at a constant angular velocity $\Delta\theta$. The NMF method provides the reconstruction $\hat{U}_{\text{NMF}} = \hat{A}_{\text{NMF}} \hat{B}_{\text{NMF}}^T$ by solving a non-negative least-square problem

$$\begin{aligned} \{\hat{A}_{\text{NMF}}, \hat{B}_{\text{NMF}}\} = \\ \arg \min_{A \geq 0, B \geq 0} \sum_{c=1}^C \|R(\theta_c) \text{vec}(AB^T) - \text{vec}(F_c)\|_2^2. \end{aligned} \quad (7.2)$$

Here the inequalities are element-wise, $R(\theta_c)$ is the system matrix approximating (Equation 7.1), and $F_c \in \mathbb{R}^{P \times N}$ records the projection collected by the c -th camera, whose (i, j) -th entry is the data on the i -th bin of the collimator at j -th frame.

7.3 Proposed Model

When the dPET radioactivity maps are observed sequentially, it is natural to identify homogeneous regions characterized by similarly varying TACs. Motivated by this observation, for a fixed number of bases $K \geq 1$ and number of iteration $I \geq 1$, we propose our model, STIS, which synthesizes temporal and spatial information for dPET reconstruction \hat{U}_{STIS} based on C -camera projection of some unknown activity map U as follows

$$\hat{U}_{\text{STIS}} = A^{(I)} (B^{(I)})^T, \quad (7.3)$$

where

$$A^{(i)} = \mathcal{N}_2 \left(\mathcal{N}_1 \left(A^{(i-1)} (B^{(i-1)})^T; \Theta_1^{(i)} \right); \Theta_2^{(i)} \right)$$

$$B^{(i)} = \arg \min_{B \in \mathbb{R}_+^{N \times K}} \sum_{c=1}^C \|R(\theta_c) \text{vec}(A^{(i)} B^T) - \text{vec}(F_c)\|_2^2$$

for $i = 1, 2, \dots, I$, and $A^{(0)} = \hat{A}_{\text{NMF}}$, $B^{(0)} = \hat{B}_{\text{NMF}}$. During the i -th iteration, the spatial basis $A^{(i)}$ is obtained by composing two networks: *TAC feature extractor* $\mathcal{N}_1(\cdot, \Theta_1^{(i)}) : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{M \times K'}$ for some integer K' such that $1 \leq K \leq K' < N$ and *domain synthesizer* $\mathcal{N}_2(\cdot, \Theta_2^{(i)}) : \mathbb{R}^{M \times K'} \rightarrow \mathbb{R}^{M \times K}$; and $\Theta_1^{(i)}$ and $\Theta_2^{(i)}$ denote the network parameters such as kernel weights and biases for the i -th components. More explicitly, the TAC feature extractor is defined for any matrix $X \in \mathbb{R}^{M \times N}$ with row vectors x_1^T, \dots, x_M^T , such that the j -th row of the output matrix $\mathcal{N}_1(X; \Theta_1^{(i)})$ is $\nu^{(i)}(x_j)$ for $j = 1, 2, \dots, M$, where $\nu^{(i)} : \mathbb{R}^N \rightarrow \mathbb{R}^{K'}$ is described in Figure 7.2 (a). The domain synthesizer is defined in Figure 7.2 (b), which maps the feature image resulted from \mathcal{N}_1 to the learned spatial bases. In this work, we fix $K' = 32$.

Our network structure is intuitive: spatial bases are naturally induced by identifying TACs sharing similar dynamic features. The temporal information plays a dominating role in our work. The TAC feature extractor maps the high dimensional and low-quality TACs in

\mathbb{R}^N to a lower-dimensional feature space. In general cases, pixels with similar TACs have similar feature representations. The method generates spatial bases to imitate the observer's impression of homogeneous regions based on intensity variation rather than the underlying biochemical functionalities or geometric affinity. Based on the resulted spatial bases, we find that applying a non-negative least-square fitting is sufficient for a stable identification of the corresponding temporal bases. The domain synthesizer is analogous to a pixel-wise image classifier. However, we note that in general image classification, an extensive range of abstract contents, such as shapes and styles, are crucial for robust recognition; hence, deeper architectures are preferable. In our setting, local affinity in the spatial domain of $\tilde{\mathcal{U}}$ is relevant to the clustering of TAC features. Therefore, we use a shallow U-net [86] to integrate the kinetic information and compose the spatial bases locally. Finally, our method provides a new and effective hybrid paradigm that has natural interpretability, enjoys data-adaptability, and allows various extensibility such as multiple iterations for refinement.

We present the following theorem to justify our motivation for using TACs to assist spatial basis reconstruction.

Theorem 7.3.1. *Let B_n denote the n -th row of the temporal basis B . For $i = 1, \dots, M$, let A_i denote the i -th row of the spatial basis A , and TAC_i be the i -th row of the radioactivity map U recording the TAC at the i -th pixel. For any $1 \leq i, j \leq M$, if there exists a positive number $\zeta > 0$ such that for all n , the ratio $\rho_n^{i,j} = \frac{\|A_i - A_j\|_2}{\|B_n\|_2}$ satisfies*

$$\begin{aligned} 1 + |\sin \theta_n| &< \rho_n^{i,j} < (3 - \zeta)/2 \\ \text{or } (1 + \zeta)/2 &< \rho_n^{i,j} < 1 - |\sin \theta_n| \end{aligned} \tag{7.4}$$

where θ_n is the angle between B_n and $A_i - A_j$, then

$$\sqrt{N\zeta} \|A_i - A_j\|_2 \leq \|TAC_i - TAC_j\|_2. \tag{7.5}$$

Proof. When i, j are fixed, to simplify the notations, we let $\gamma = A_i - A_j$ and denote

$z_n = \|\gamma\|_2 / \|B_n\|_2$. Consider the following differentiable function of $h \in \mathbb{R}$

$$f(h) = \frac{\left\| \frac{B_n}{\|B_n\|_2} - \frac{\gamma}{\|\gamma\|_2} h \right\|_2}{|1 - z_n|} - h = \frac{\sqrt{h^2 - 2h \cos \theta_n + 1}}{|1 - z_n|} - h ,$$

where $\cos \theta_n = \frac{B_n^T \gamma}{\|B_n\|_2 \|\gamma\|_2}$. Notice that f can only have positive root if any exists. By the assumption that $(3 - \zeta)/2 > z_n > 1 + |\sin \theta_n|$ or $(1 + \zeta)/2 < z_n < 1 - |\sin \theta_n|$, it is readily checked that f always has a root $h_n^+ = \frac{-\cos \theta_n - \sqrt{z_n^2 - 2z_n + \cos^2 \theta_n}}{z_n^2 - 2z_n} > 0$. Evaluating $f(h_n^+)$ and reorganizing terms gives

$$\left\| B_n - \frac{h_n^+ \|B_n\|_2}{\|\gamma\|_2} \gamma \right\|_2 = \frac{h_n^+}{2} \left| \frac{2\|B_n\|_2}{\|\gamma\|_2} - 2 \right| \|\gamma\|_2$$

Now taking $H_n = \frac{2}{z_n} - 1$, the equation above becomes

$$\left\| B_n - \frac{H_n h_n^+ + h_n^+}{2} \gamma \right\|_2 = \frac{|H_n h_n^+ - h_n^+|}{2} \|\gamma\|_2 ,$$

which allows us to apply the reverse Cauchy-Schwarz inequality [448] and gives

$$\begin{aligned} \|\text{TAC}_i - \text{TAC}_j\|_2^2 &= \sum_{n=1}^N \gamma^T B_n \geq \\ &\sum_{n=1}^N \left(\|B_n\|_2^2 - \frac{1}{4} \|\gamma\|_2^2 ((H_n - 1) h_n^+)^2 \right) \|\gamma\|_2^2 . \end{aligned} \quad (7.6)$$

We then prove that $\|B_n\|_2^2 > \frac{1}{4} \|\gamma\|_2^2 ((H_n - 1) h_n^+)^2 + \zeta$, hence Equation 7.6 is useful. Consider the case where $1 < 1 + |\sin \theta_n| < z_n < (3 - \zeta)/2 < 2$, indicating that $\zeta z_n^2 (z_n - 2)^2 + 2z_n - 3 < 0$, which can be written as $\frac{z_n^2 (z_n - 1)^2}{(z_n^2 - 2z_n)^2} + \zeta z_n^2 < 1$. Notice that this implies $(1 - z_n)^2 (h_n^+)^2 + \zeta z_n^2 < 1$, which by simple computation leads to $\|B_n\|_2^2 > \frac{1}{4} \|\gamma\|_2^2 ((H_n - 1) h_n^+)^2 + \zeta$. Similarly, we can prove for the case where $1 > 1 - |\sin \theta_n| >$

$z_n > (1 + \zeta)/2$ by starting with $\zeta z_n^4 - 2z_n + 1 < 0$. Therefore, from Equation 7.6, we have

$$\|\text{TAC}_i - \text{TAC}_j\|_2^2 \geq N\zeta\|\gamma\|_2^2.$$

which completes the proof. \square

Theorem 7.3.1 implies that, when the sine distance between B_n and $A_i - A_j$ is small, or equivalently, two vectors are highly correlated, and their scales are comparable, if the i, j pixels have similar TACs, their spatial bases at these positions need to have close values. In particular, if we interpret the value $A_{i,k}$ as the belief of the i -th pixel belonging to the k -th basis, Theorem 7.3.1 says that pixels can be classified by utilizing their TACs.

7.4 Numerical Experiments

We present different numerical results to validate our model. Our dataset consists of 192 synthetic images from [442]. Each sample contains 90 frames of 64×64 images, where the pixel intensity variations were induced by TAC curves distributed over an anatomic mask of a rat’s abdomen. The system matrix simulates a two-head camera which projects to collimators with 95 bins at two orthogonal angles per frame. Initially fixed at the top and right of the domain of interest, these cameras rotate 1° per frame clockwise. To optimize the network parameters, we used 80% of the data for training and the rest for testing. We used the Adam optimizer with a learning rate 5×10^{-4} , which is factored by 0.8 every 50 epochs. The training consists of 500 epochs in total. In all the experiments, we fix the decay rate $\mu = 0.01$. Results shown in this paper are from the testing dataset. We trained our model using Intel(R) Xeon(R) CPU E5-2689 v4 @ 3.10GHz with 10 cores and 20 threads, equipped with 24G RTX Titan V GPU.

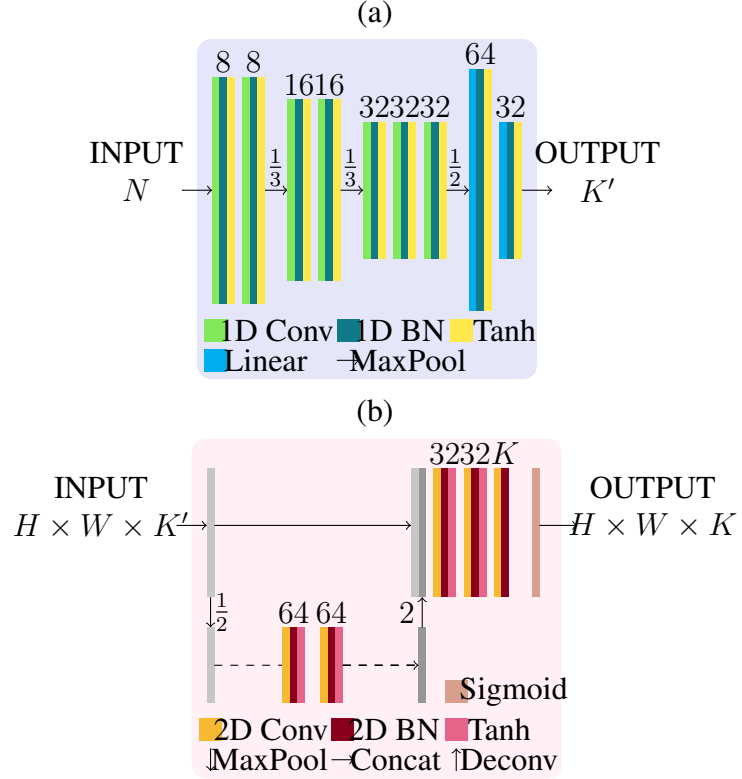


Figure 7.2: (a) Structure of a coordinate component of the TAC feature extractor \mathcal{N}_1 . The input is a TAC in \mathbb{R}^N ($N = 90$) and the output is a feature vector in $\mathbb{R}^{K'}$ ($K' = 32$). The convolution kernels are of size 3 and the zero-paddings are of size 1. The numbers above Conv are the number of channels, and those above Linear are node sizes. (b) Structure of the domain synthesizer \mathcal{N}_2 . The input is an image whose channels are TAC feature vectors. Here H is the image height and W is the image width such that $M = HW$. The output is an image with K channels, each of which defines a spatial basis for the reconstruction.

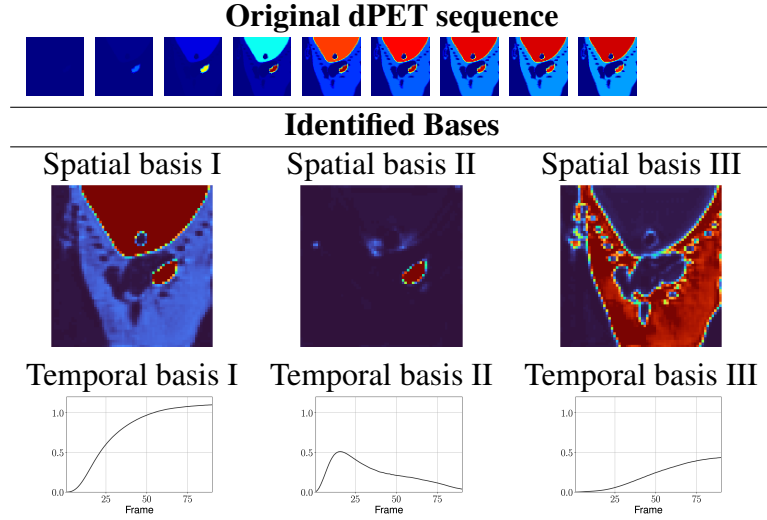


Figure 7.3: Interpretability of spatial and temporal bases identified by STIS using a low rank ($K = 3$). Each spatial basis roughly corresponds to homogeneous regions with similar dynamic features, and the associated temporal basis describes its contribution to the concentration distribution.

7.4.1 Interpretability of Low-rank Bases

When K is set at a sufficient level to cover the underlying radiotracer concentration, the proposed STIS model produces low-rank dPET reconstruction with interpretable bases. With $K = 3$ and $I = 2$, Figure Figure 7.3 shows an original dPET sequence in the first row and the identified spatial and temporal bases in the second and third row, respectively. STIS generates spatial bases as homogeneous regions with similar radioactivity variations. For example, the 1st basis corresponds to the high concentration regions. The tumor, which rapidly accumulates tracers in the early frames is captured by the 2nd basis. The 3rd basis characterizes the tissue with a slower accumulation rate and lower stabilizing level. Correspondingly, each temporal basis defines the contribution of each spatial basis to the activity map.

7.4.2 Performance on TAC Reconstruction

As an indicator of the radiotracer concentration measured over time, the TAC plays an essential role in pharmacokinetic analysis and clinical diagnosis. For instance, the slope of

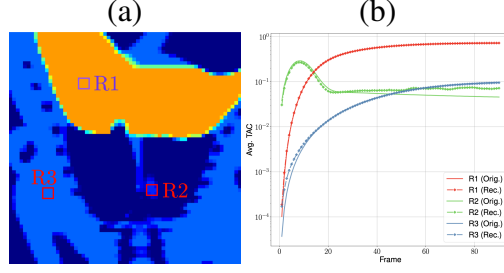


Figure 7.4: Performance on TAC reconstruction. (a) Three 3×3 square ROIs marked on the final frame of an original test sample. (b) Average TACs of the respective ROIs in (a).

TAC can be helpful in non-invasive discrimination of brain tumor subtype [449]. Focusing on a test sample and its three ROIs showed in Figure 7.4(a) and using STIS with $K = 3$, $I = 2$, we recorded the respective reconstructed TACs and the original ones in (b). In this experiment, each ROI is a 3×3 square, and the TACs were averaged within the corresponding regions. For the rapidly accumulating region with a steady high concentration (R1), STIS recovered the TACs with high accuracy. For the region with a moderate accumulation rate and a lower stabilized level (R2), the reconstructed TACs stay close to the true ones, especially after frame 20. For the region where the radiotracer’s concentration rapidly grows and decays (R3), STIS successfully captured this dynamic feature of the underlying TACs and correctly estimated the peak level and the peak width.

7.4.3 Tests on Hyperparameters

There are two major hyperparameters in the proposed model, i.e., the number of bases (K), and the number of iterations (I). On the positive side, STIS with a larger K has higher flexibility and better generalizability; and STIS with a larger I can further refine the reconstruction results. However, increasing K and I makes the training unstable and more demanding to converge. In Table 7.1, we quantitatively compare STIS using various combinations of K and I by the average RMSE of reconstruction based on the testing dataset. First, we see that STIS models with $I \geq 2$ have a clear advantage over those with $I = 1$. This was expected, since the spatial bases identified by STIS with $I = 1$ do not gain

Table 7.1: Comparison among proposed model with various combinations of hyperparameters: number of bases (K) and number of iterations (I).

Config.	RMSE	Config.	RMSE	Config.	RMSE
K3I1	3.55e-2	K3I2	2.41e-2	K3I3	2.42e-2
K5I1	2.82e-2	K5I2	2.11e-2	K5I3	2.32e-2
K7I1	2.71e-2	K7I2	2.19e-2	K7I3	2.10e-2

any improvement resulted from the updated temporal bases. Second, except for K7I2, we observe a general improvement of using larger K when I is fixed. Third, STIS performs better when both K and I increase coordinately. It means that simply increasing either I or K while the other fixed may not lead to better results. To cope with the training instability when I is greater than 3, other network structure modifications are necessary.

7.4.4 Qualitative Comparison

We compare the proposed model, STIS, with some methods in the literature from the visual perspective. Specifically, we consider two model-based approaches, NMF (Equation 7.2) and SEMF [450], and two data-driven approaches, DnCNN [451](image domain denoising method) and LEARN [452] (optimization-unrolling-based reconstruction method). For NMF, we fix the number of bases as 7 and apply the oblique projected Landweber algorithm [453] with 100 outer iterations and 5 inner iterations. For SEMF, the iteration number is fixed to be 50 as the convergence is observed and the base number 7 (SEMF7) and 16 (SEMF16) are tested. Both DnCNN and LEARN are trained for 200 epochs on our training dataset. Taking NMF result as input, DnCNN with 17 layers is trained for 3D image denoising. For LEARN, we consider 50 iterations in the unrolling model and the other network parameters are the same as [452].

On a projection input data chosen from the test dataset, we added 1% Gaussian noise and tested these methods whose results are shown in Figure 7.5. As expected, without regularization, NMF suffered from blurriness in the face of noise and sparse sampling. DnCNN roughly captured the intensity variation trend and approximately recovered the

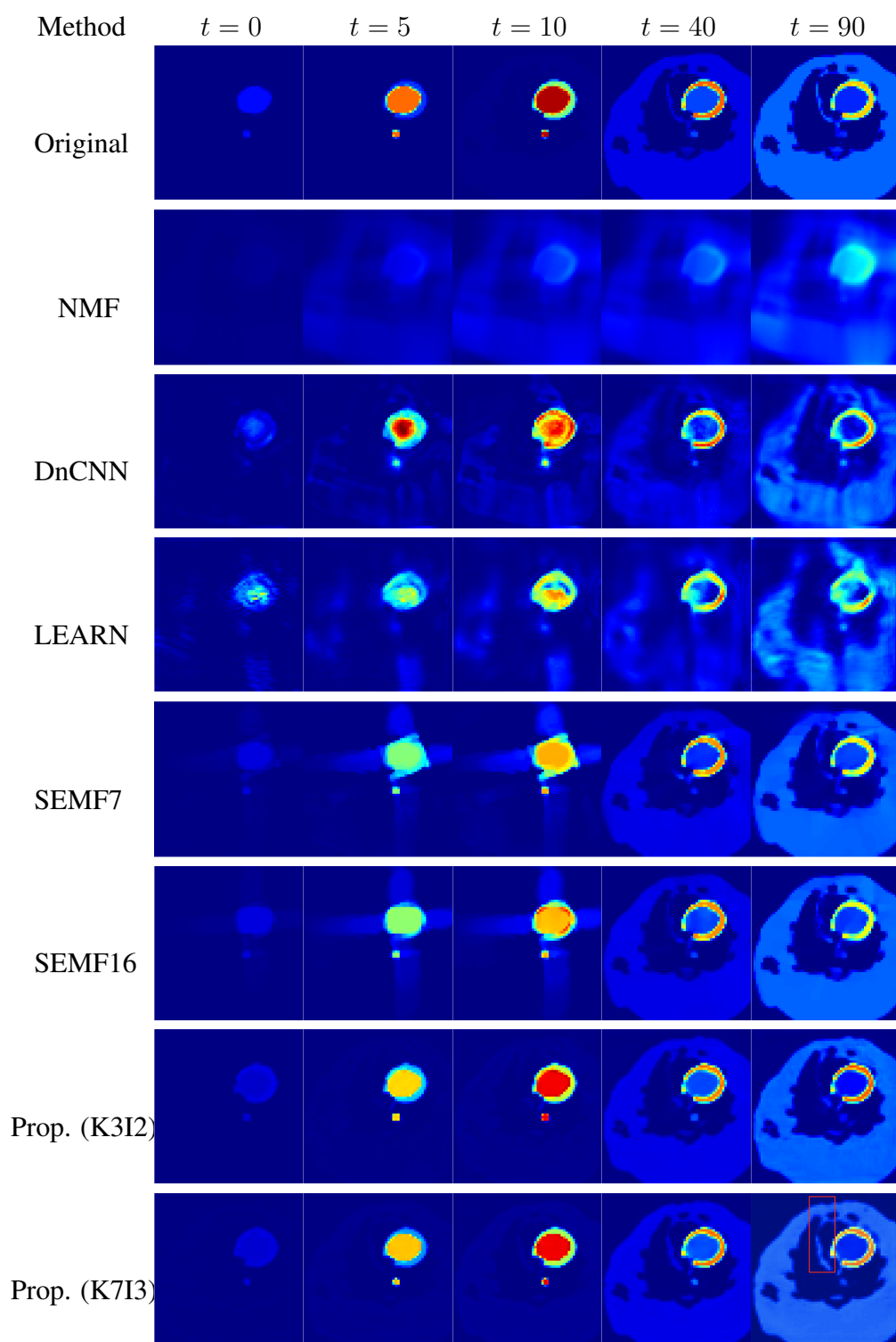


Figure 7.5: Qualitative comparison of different methods on a test dPET image sequence. The mark region is further examined in Figure 7.6.

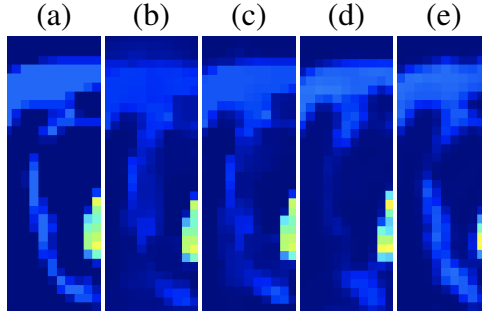


Figure 7.6: Zoom-in comparisons among (a) Original (b) SEMF7 (c) SEMF16 (d) Proposed (K3I2) (e) Proposed (K7I3) on the final frame of the tests in Figure 7.5.

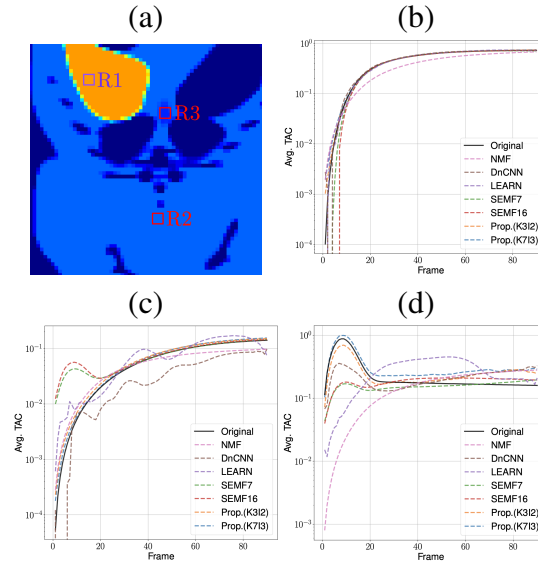


Figure 7.7: Comparison of different methods on TAC reconstruction. (a) Three 3×3 square ROIs marked on the final frame of an original test sample. (b) Comparison of the average TACs in R1 (c) in R2, and (d) in R3.

regions with high concentration level; however, many geometric details of the underlying organs were lost. Similarly to DnCNN, LEARN produced false ring patterns due to sparse sampling, and the reconstructed frames for the earlier frames are unstable. Both SEMF7 and SEMF16 produced satisfying reconstructions for the later frames where TACs vary slowly, and compared to SEMF7, boundaries rendered by SEMF16 become sharper. Their reconstructions for earlier frames are less appealing. Observe that there are significant shadowing artifacts projecting from the identified rapidly glowing regions, whose levels of concentration are not comparable to the underlying ones. Our proposed method using either configuration K3I2 and K7I3 successfully recovered both the kinetic and geometric features of the underlying dPET image sequence. Notice that both K3I2 and K7I3 have superior performance for the reconstruction during the early stage, immune from prominent artifacts. With larger numbers of bases and iterations, K7I3 greatly improved the mild oscillation in the homogeneous regions and loss of fine details by K3I2. This is further justified by the zoom-in comparison in Figure 7.6, where we also observe that the delicate tissue recovered by K7I3 has better continuity and a more accurate level of concentration compared to either SEMF7 or SEMF16.

Figure 7.7 compares different methods in terms of TAC reconstructions in three ROIs marked in a final frame of a test sample in (a). For the region with a rapid increment and high stabilizing level (R1), all methods have comparable approximations of the underlying truth. In R2, where the radiotracer concentration grows slowly and converges to a relatively lower level, TACs reconstructed by NMF and the proposed method with K3I2 and K7I3 stay close to the underlying one. Although both SEMF7 and SEMF16 stay around the correct level, they present oscillatory behaviors. This is analogous to the reconstruction from DnCNN. The concentration in R3 grows first then decays rapidly and converges to a relatively low level. Both LEARN and NMF fail to capture the concentration peak. DnCNN, SEMF7 and SEMF16 detect the existence of peak, yet the peak levels are far from the true value. The proposed method has superior identification of the peak height

and width.

7.4.5 Quantitative Comparison

For further validation, we quantitatively compared the above methods by measuring the reconstruction quality using RMSE, PSNR, and SSIM in Table 7.2 and evaluating their efficiency in Table 7.3 based on the same testing dataset. In particular, for the reconstruction quality, we evaluate the results grouped by three different ranges of frames: (i) $1 \sim 5$, (ii) $5 \sim 15$, and (iii) $15 \sim 90$, which roughly correspond to (i) the early stage where the concentration of radiotracer is generally low, (ii) the active stage where abrupt increment or decay of concentration mostly occurs, and (iii) the final stage where most TACs evolve slowly or stabilize. For examining the efficiency of data-driven methods, we recorded both training time and testing time for a fair comparison.

Consistent with the visual inspection, NMF without any regularization performs poorly. Both DnCNN and LEARN produce better quality in stage (i), and the reconstructions are unstable in stage (ii) and (iii) in general. In terms of reconstruction quality, the most competitive models are SEMF and STIS. We note that the results by SEMF enjoy higher quality than those by STIS in stage (iii) using all these metrics. Based on PSNR, SEMF also outperforms STIS in stage (ii). However, both RMSE and SSIM, which are less controversial than PSNR as video quality metrics [454], indicate that STIS provides superior reconstructions over SEMF in stage (i) and (ii). The variability of TACs during these stages is pertinent to kinetic parameters such as the tracer exchange rates [455]. Hence, the quality of reconstruction of these frames is critical in practice. Moreover, when considering the efficiency, reconstructing a dPET sequence takes about 21.40 minutes for SEMF7, whereas it only takes the trained STIS (K7I3) about 5.09 seconds to produce the high-quality results.

Table 7.2: Quantitative comparison (Mean \pm Std.) of different methods’ performances on the testing dataset (39 samples). For the results of quality evaluation, the **best** ones are bolded, and the second best* ones are marked with asterisks. Deep learning based methods were trained using the same training dataset (153 samples) on a common machine configuration.

Index	Frame	NMF	DnCNN	LEARN	SEMF7	SEMF16	Prop.(K3I2)	Prop.(K7I3)
RMSE $\times 10^{-2}$	1 \sim 5	2.19 \pm 2.97	1.91 \pm 1.60	2.21 \pm 1.93	1.62 \pm 1.94	1.64 \pm 2.04	1.38 \pm 1.62	1.41 \pm 1.45*
	5 \sim 15	3.19 \pm 3.79	2.06 \pm 2.12	2.51 \pm 2.53	1.71 \pm 2.31	2.67 \pm 3.81	1.43 \pm 1.84*	1.41 \pm 1.78
	15 \sim 90	8.24 \pm 4.93	3.19 \pm 1.82	5.34 \pm 3.31	1.32 \pm 1.06*	1.05 \pm 0.90	2.11 \pm 1.44	1.80 \pm 1.19
PSNR $\times 10$	1 \sim 5	4.58 \pm 1.95	3.71 \pm 0.68	3.62 \pm 0.73	4.84 \pm 1.96	5.30 \pm 2.45	4.92 \pm 1.90*	4.24 \pm 1.05
	5 \sim 15	2.06 \pm 1.56	2.23 \pm 1.59	2.10 \pm 1.50	2.73 \pm 2.20*	2.93 \pm 2.46	2.69 \pm 2.04	2.64 \pm 1.96
	15 \sim 90	1.73 \pm 0.84	2.40 \pm 1.10	2.04 \pm 0.97	3.12 \pm 1.47*	3.31 \pm 1.57	2.74 \pm 1.28	2.84 \pm 1.32
SSIM	1 \sim 5	0.89 \pm 0.18	0.89 \pm 0.05	0.83 \pm 0.08	0.91 \pm 0.10	0.91 \pm 0.10	0.96 \pm 0.05	0.95 \pm 0.05*
	5 \sim 15	0.50 \pm 0.36	0.57 \pm 0.39	0.51 \pm 0.35	0.61 \pm 0.42	0.61 \pm 0.43	0.66 \pm 0.45	0.66 \pm 0.45
	15 \sim 90	0.45 \pm 0.22	0.64 \pm 0.29	0.51 \pm 0.24	0.80 \pm 0.36*	0.81 \pm 0.36	0.76 \pm 0.35	0.78 \pm 0.35

Table 7.3: Training and testing efficiency.

Stage	NMF	DnCNN	LEARN	SEMF7	SEMF16	Prop.(K3I2)	Prop.(K7I3)
Training Time	-	5.5 hour	3.1 days	-	-	1.3 days	1.9 days
Testing Time	4.83 min.	0.16 sec.	2.39 sec.	21.40 min.	32.50 min.	2.60 sec.	5.09 sec.

7.5 Summary

In this chapter, we described the STIS model, which synthesizes temporal and spatial information for reconstructing the dPET activity maps from sparsely sampled projections. There are a limited number of work for dPET reconstruction under the comparable sampling condition, where only two projection angles are available for each frame. By our novel TAC feature extractor and domain synthesizer, the proposed STIS is robust against deficient samples, and it also renders interpretable spatial and temporal bases for the low-rank reconstruction of dPET sequences. By comparison studies, we observed that STIS preserves homogeneous regions and definite boundaries. It captures kinetic features such as abrupt increment and decay of radiotracer concentration, which are generally challenging for the other methods.

CHAPTER 8

CONCLUSION

This thesis showcases diverse types of mathematical pattern representation and illustrates their applications in various fields. We started with the rigid pattern of lattices represented by pairs of complex numbers in a quotient space, whose equivalence relations are characterized by the modular group and metric structure is derived from the Poincaré geometry. This lattice metric space encodes not only the defining properties of every lattice pattern, it also allows measurement of the visual differences among any two lattices. Then we dived into a more flexible visual pattern: shapes. Unlike symmetries in rigid patterns, human perception plays a more critical role. As a consequence, the description of shape patterns is less structural. From region-based and contour-based perspectives, we discussed two shape representations with different understanding of how human perceives and distinguishes shape patterns. To reflect the scale-invariant geometric features deemed as visual cues, we utilized the affine-shortening PDE to preserve prominent characteristics in pattern recognition. With more versatile data distributions, identifying suitable submanifold representations of point clouds starts to reflect the importance of regularization, which imposes model's stability against data noise and enforces desired representation properties. Meanwhile, we discussed the fast algorithms associated with the resulted non-convex optimization problems. As moving on to the next topic, underwater color correction, we introduced a model-based approach: Tikhonov optimization framework in CIELAB motivated by the complementary adaptation theory in psychology, yet the color representation already shows a strong dependence on experimental data. For example, the correction of the blue region uniformity and the adjustment for Helmholtz-Kohlrausch effect involve many parameter fittings. This discussion marks a critical point of transitioning from model-based approaches to data-driven ones. For the data-driven representations, we started with

the automatic PDE modeling from noisy data which illustrates how to identify suitable PDE representations by learning spatial-temporal features of single trajectories. This project reveals the influence of data cast on the model selection. Given a fixed underlying PDE, the data sampled from some trajectories may lead to more accurate identification than the others. Theoretically, we also discussed the role of sparsity from the point of view of statistical sparse signal recovery and proved that the signed support of the underlying PDE features can be recovered if certain conditions are satisfied by the sample data. Finally, we entered the topic of deep learning by focusing on an application in medical image reconstruction. Instead of an end-to-end training paradigm, we described an interesting network structure based on a low-rank model-based representation of the reconstructed image sequences. By this hybrid scheme, we gain a considerable dimension reduction on the feature space, thus leading to a more stable training.

In general, for rigid patterns, the mathematical framework is more definite, and the analysis on the model can directly lead to knowledge about the pattern. As for more versatile patterns, it is necessary to supplement reasonable assumptions for an accessible model-based representation due to the complicated or unclear mechanisms behind the pattern formation and recognition. From different perspectives, we can introduce various regularization terms or priors to specify desired properties about the solutions and manually tune the associated parameters to obtain good results. On the positive side, most mathematically defined regularization terms bear clear indication of the enforced conditions, i.e., smoothness, and piecewise constant, thus by adjusting the parameters, we foresee what are the effects on the results. However, many challenging inverse problems require multiple regularizations, and choosing which combinations as well as the optimal parameters is a rather involved task. Data-driven approaches, on the other hand, address such challenges by learning pattern features from data. With sufficiently many data for training, DNNs often acquire powerful expressivity such that they can approximate large classes of mappings. When the training data is not enough, end-to-end learning can fail and the training process

is unstable. A promising direction is to look into hybrid scheme where model-based approaches can be applied to establish the basic framework, while leaving the unknown or complicated components learned from data. Therefore, the training only needs to focus on learning limited number of features, and the fundamental relations among samples are specified by well-defined models.

Data-driven representations, especially those constructed by deep learning methods, have been extensively explored in recent years; however, this by no means indicates that model-based approaches are outdated. For well-studied patterns, model-based representations provide accurate description, and many more implications can be derived via rigorous analysis. In other words, model-based methods are white-boxes. In contrast, data-driven representations yield mappings that reflect various features of the available data, yet it is demanding to interpret the resulted network; hence data-driven methods are often black-boxes. There is no universal criteria to decide which approach is absolutely better than the other. In fact, in some cases, model-based representations noticeably outperform the data-driven ones, and such situation occurs especially when the training data is expensive to access with large amount. It is thus important and beneficial to keep open-minded about both directions.

Appendices

APPENDIX A

APPENDIX FOR CHAPTER 2

A.1 Psudo-code for computing $d_{\mathcal{L}}$

The definition of $d_{\mathcal{L}}$ (Equation 2.11) requires multiple comparisons. For paths passing through $\{(\beta, \rho) \mid \beta \in \mathcal{K}, |\rho| = 1, \rho \in \mathcal{P}\}$, the minimal is found by considering all four paths.

Inputs: two lattice bases (b_1, b_2) and $(b'_1, b'_2) \in \mathbb{C}^2$.

Step 1. Transfer to descriptors: $\beta \leftarrow b_1, \beta' \leftarrow b'_1, \rho \leftarrow b_2/b_1$, and $\rho' \leftarrow b'_2/b'_1$.

Step 2. Compute $D((\beta, \rho), (\beta', \rho'))$ as defined in (Equation 2.8).

Step 3. Fix an integer N .

For $j = 0, 1, \dots, N$:

For $k = 0, 1, \dots, N$:

$$D_{j,k} \leftarrow D(\beta, \rho', \beta, \rho');$$

$$D_{j,k} \leftarrow \min\{D_{j,k}, D(\beta, \rho, e^{i(\pi/3+k\pi/3)}\beta', -1/e^{i(\pi/3+k\pi/3)}) + D(\beta', e^{i(\pi/3+k\pi/3)}, \beta', \rho')\};$$

$$D_{j,k} \leftarrow \min\{D_{j,k}, D(\beta, \rho, \beta, e^{i(\pi/3+j\pi/3)}) + D(e^{i(\pi/3+j\pi/3)}\beta, -1/e^{i(\pi/3+j\pi/3)}, \beta', \rho')\};$$

$$D_{j,k} \leftarrow \min\{D_{j,k}, D(\beta, \rho, \beta, e^{i(\pi/3+j\pi/3)}) + D(e^{i(\pi/3+j\pi/3)}\beta, -1/e^{i(\pi/3+j\pi/3)}, \dots \\ e^{i(\pi/3+j\pi/3)}\beta', -1/e^{i(\pi/3+j\pi/3)}) + D(\beta', e^{i(\pi/3+j\pi/3)}, \beta', \rho')\};$$

End For

End For

$$d_{\mathcal{L}}((\beta, \rho), (\beta', \rho')) \leftarrow \min_{j,k} D_{j,k}.$$

APPENDIX B

APPENDIX FOR CHAPTER 3

B.1 Dataset and Image Credits for Shape Skeleton



7 Core Experiment CE-Shape-1 Test Set¹ [456], provided by Dr. Longin Jan Latecki, Professor, Department of Computer and Information Sciences, Temple University, US

 ² Cat Stretch Silhouette In Black, CC0 Public Domain K

 ³  ⁴  ⁵ svgsilh.com, Creative Commons CC0

B.2 Silhouette Data Set

In Table Table B.1, we collectively display the 20 silhouettes used in this paper. They are all downloadable from <https://svgsilh.com>, which are released under Creative Commons CC0.

¹<http://www.dabi.temple.edu/~shape/MPEG7/dataset.html>

²<https://www.publicdomainpictures.net/en/view-image.php?image=32201&picture=cat-stretch-silhouette-in-black>

³<https://svgsilh.com/image/152115.html>

⁴<https://svgsilh.com/image/365843.html>

⁵<https://svgsilh.com/image/1614530.html>

Table B.1: Silhouette dataset used in the experiments. The last four are used in Figure 3.13 for computing the average $\rho(\tau_e)$. These silhouettes are chosen from [253], which are released under Creative Commons CC0.



APPENDIX C
APPENDIX FOR CHAPTER 6

C.1 Proof of Proposition Theorem 6.2.1

Proof.

$$\begin{aligned}
& [D_t U]^{\mathcal{T}_2} - [F]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [D_t U]^{\mathcal{T}_1} \\
&= [D_t U]^{\mathcal{T}_2} - [u_t]^{\mathcal{T}_2} + [u_t]^{\mathcal{T}_2} - [F]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [D_t U]^{\mathcal{T}_1} \\
&= \underbrace{[D_t U]^{\mathcal{T}_2} - [u_t]^{\mathcal{T}_2}}_{E_1} + [u_t]^{\mathcal{T}_2} - [F]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [u_t]^{\mathcal{T}_1} - \underbrace{[F]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger ([D_t U]^{\mathcal{T}_1} - [u_t]^{\mathcal{T}_1})}_{E_2} \\
&= [u_t]^{\mathcal{T}_2} - ([F_0]_{\mathcal{A}}^{\mathcal{T}_2} + [F]_{\mathcal{A}}^{\mathcal{T}_2} - [F_0]_{\mathcal{A}}^{\mathcal{T}_2}) ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [u_t]^{\mathcal{T}_1} + E_1 + E_2 \\
&= [u_t]^{\mathcal{T}_2} - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [u_t]^{\mathcal{T}_1} - \underbrace{([F]_{\mathcal{A}}^{\mathcal{T}_2} - [F_0]_{\mathcal{A}}^{\mathcal{T}_2}) ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger [u_t]^{\mathcal{T}_1}}_{E_3} + E_1 + E_2 \\
&= \underbrace{[u_t]^{\mathcal{T}_2} - [F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger [u_t]^{\mathcal{T}_1}}_{=0} + ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1} \\
&+ E_1 + E_2 + E_3 \\
&= ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1} + E_1 + E_2 + E_3 \\
&= ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1} \\
&- \underbrace{[F_0]_{\mathcal{A}}^{\mathcal{T}_2} (([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger - ([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1}}_{E_4} + E_1 + E_2 + E_3 \\
&= ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1} + E_1 + E_2 + E_3 + E_4 .
\end{aligned}$$

Then we have:

$$\begin{aligned}
\text{CEE}(\mathcal{A}_k; \alpha, \mathcal{T}_1, \mathcal{T}_2) &\leq \|([F_0]_{\mathcal{A}_0}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}_0}^{\mathcal{T}_1})^\dagger - [F_0]_{\mathcal{A}}^{\mathcal{T}_2} ([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger) [u_t]^{\mathcal{T}_1}\|_2 \\
&+ \|[D_t U]^{\mathcal{T}_2} - [u_t]^{\mathcal{T}_2}\|_2 + \|([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 (\|[F]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|[D_t U]^{\mathcal{T}_1} - [u_t]^{\mathcal{T}_1}\|_2 \\
&+ \|[F]_{\mathcal{A}}^{\mathcal{T}_2} - [F_0]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|[u_t]^{\mathcal{T}_1}\|_2) \\
&+ \|[F_0]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 \|([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 \|[F]_{\mathcal{A}}^{\mathcal{T}_1} - [F_0]_{\mathcal{A}}^{\mathcal{T}_1}\|_2 \|[u_t]^{\mathcal{T}_1}\|_2 .
\end{aligned}$$

In the last term on the right hand side of the inequality, we applied the norm bound in Theorem 4.1 of [457]. Then by setting

$$\begin{aligned}
g(\mathcal{A}; \alpha, \mathcal{T}_1, \mathcal{T}_2) &= \|[D_t U]^{\mathcal{T}_2} - [u_t]^{\mathcal{T}_2}\|_2 + \|([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 (\|[F]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|[D_t U]^{\mathcal{T}_1} - [u_t]^{\mathcal{T}_1}\|_2 \\
&+ \|[F]_{\mathcal{A}}^{\mathcal{T}_2} - [F_0]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|[u_t]^{\mathcal{T}_1}\|_2) \\
&+ \|[F_0]_{\mathcal{A}}^{\mathcal{T}_2}\|_2 \|([F]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 \|([F_0]_{\mathcal{A}}^{\mathcal{T}_1})^\dagger\|_2 \|[F]_{\mathcal{A}}^{\mathcal{T}_1} - [F_0]_{\mathcal{A}}^{\mathcal{T}_1}\|_2 \|[u_t]^{\mathcal{T}_1}\|_2 \quad (\text{C.1})
\end{aligned}$$

we have proved the theorem. □

C.2 Primal-Dual Witness construction

In this Section, we briefly review the PDW construction in [458] for reader's convenience. A primal-dual pair $(\hat{\beta}, \hat{z}) \in \mathbb{R}^{K \times K}$ is said to be optimal if $\hat{\beta}$ is a minimizer of (Equation 6.34) and $\hat{z} \in \partial \|\hat{\beta}\|_1$, where $\partial \|\hat{\beta}\|_1$ denotes a sub-differential set of $\|\cdot\|_1$ evaluated at $\hat{\beta}$. Any such pair must satisfy zero-subgradient condition of (Equation 6.34), which is as follows:

$$-\frac{1}{NM} \hat{\mathbf{F}}^T (\hat{\mathbf{u}}_t - \hat{\mathbf{F}} \hat{\beta}) + \lambda \hat{z} = 0, \text{ for } \hat{z} \in \partial \|\hat{\beta}\|_1. \quad (\text{C.2})$$

Recall that we denote the ground-truth support of β^* as \mathcal{S} , and suppose that we know \mathcal{S} apriori. For the ground-truth support set \mathcal{S} and its complement set \mathcal{S}^c , PDW is said to be successful if the constructed tuple, $(\hat{\beta}_{\mathcal{S}}, \hat{\beta}_{\mathcal{S}^c}, \hat{z}_{\mathcal{S}}, \hat{z}_{\mathcal{S}^c})$, is primal-dual optimal, and act as a witness for the fact that the LASSO finds the unique optimal solution with correct support set. We construct the tuple through the following three steps.

1. Set $\hat{\beta}_{\mathcal{S}^c} = 0$.
2. Find $(\hat{\beta}_{\mathcal{S}}, \hat{z}_{\mathcal{S}})$ by solving the s -dimensional oracle sub-problem

$$\hat{\beta}_{\mathcal{S}} \in \arg \min_{\beta_{\mathcal{S}} \in \mathbb{R}^s} \left\{ \frac{1}{2NM} \left\| \hat{\mathbf{u}}_t - \hat{\mathbf{F}}_{\mathcal{S}} \beta_{\mathcal{S}} \right\|_2 + \lambda \|\beta_{\mathcal{S}}\|_1 \right\},$$

where s is the cardinality of the set \mathcal{S} . Thus $\hat{z}_{\mathcal{S}} \in \partial \|\hat{\beta}_{\mathcal{S}}\|_1$ satisfies the relation $-\frac{1}{NM} \hat{\mathbf{F}}_{\mathcal{S}}^T (\hat{\mathbf{u}}_t - \hat{\mathbf{F}}_{\mathcal{S}} \hat{\beta}_{\mathcal{S}}) + \lambda \hat{z}_{\mathcal{S}} = 0$.

3. Solve for $\hat{z}_{\mathcal{S}^c}$ through the zero-subgradient equation (Equation C.2), and check whether or not the *strict dual feasibility* condition $\|\hat{z}_{\mathcal{S}}\|_{\infty} < 1$ holds.

C.3 Local-Polynomial estimator : Closed-form solutions

Recall that we want to solve following two optimization problems for constructing $\hat{\mathbf{u}}_t$ and $\hat{\mathbf{F}}$, given the noisy observation $\mathcal{D} = \{(X_i, t_n, U_i^n) \mid i = 0, \dots, M-1; n = 0, \dots, N-1\}$.

$$\left\{ \hat{b}_j(X_i, t) \right\}_{j=0,1,2} = \arg \min_{b_j(t) \in \mathbb{R}, 0 \leq j \leq 2} \sum_{n=0}^{N-1} \left(U_i^n - \sum_{j=0}^2 b_j(t) (t_n - t)^j \right)^2 \mathcal{K}_{h_N} \left(t_n - t \right),$$

for $i = 0, 1, \dots, M-1$; (C.3)

$$\left\{ \hat{c}_j^p(x, t_n) \right\}_{j=0,1,\dots,p+1} = \arg \min_{c_j(t) \in \mathbb{R}, 0 \leq j \leq p+1} \sum_{i=0}^{M-1} \left(U_i^n - \sum_{j=0}^{p+1} c_j^p(t) (X_i - x)^j \right)^2 \mathcal{K}_{w_M} \left(X_i - x \right)$$

for $n = 0, 1, \dots, N-1$ and $p = 0, 1, \dots, P_{\max}$. (C.4)

and set $\widehat{u}_t(X_i, t) = \widehat{b}_1(X_i, t)$ and $\widehat{\partial_x^p u}(x, t_n) = p! \widehat{c}_p^p(x, t_n)$. Then, the standard weighted least-square theory leads to the solutions of (Equation C.3) and (Equation C.4), respectively:

$$\widehat{u}_t(X_i, t) = \xi_1^T (\mathbf{T}_1^T \mathbf{W}_t \mathbf{T}_1)^{-1} \mathbf{T}_1^T \mathbf{W}_t \mathbf{U}_i, \quad \forall i = 0, 1, \dots, M-1, \quad (\text{C.5})$$

$$\widehat{\partial_x^p u}(x, t_n) = p! \xi_{p,x}^T (\mathbf{X}_p^T \mathbf{W}_x \mathbf{X}_p)^{-1} \mathbf{X}_p^T \mathbf{W}_x \mathbf{U}^n, \quad \forall p = 0, 1, \dots, P_{\max}, \quad \forall n = 0, 1, \dots, N-1, \quad (\text{C.6})$$

where $\mathbf{U}_i = [U_i^0, \dots, U_i^{N-1}]^T$ and $\mathbf{U}^n = [U_0^n, \dots, U_{M-1}^n]^T$, and

$$\mathbf{T}_1 := \begin{bmatrix} 1 & t_0 - t & (t_0 - t)^2 \\ 1 & t_1 - t & (t_1 - t)^2 \\ \vdots & \vdots & \vdots \\ 1 & t_{N-1} - t & (t_{N-1} - t)^2 \end{bmatrix}, \quad \mathbf{X}_p := \begin{bmatrix} 1 & X_0 - x & \dots & (X_0 - x)^{p+1} \\ 1 & X_1 - x & \dots & (X_1 - x)^{p+1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & X_{M-1} - x & \dots & (X_{M-1} - x)^{p+1} \end{bmatrix},$$

for $p = 0, \dots, P_{\max}$, and

$$\mathbf{W}_t := \text{diag}\{\mathcal{K}_{h_N}(t_0 - t), \dots, \mathcal{K}_{h_N}(t_{N-1} - t)\},$$

$$\mathbf{W}_x := \text{diag}\{\mathcal{K}_{w_M}(X_0 - x), \dots, \mathcal{K}_{w_M}(X_{M-1} - x)\},$$

are $N \times N$ and $M \times M$ diagonal matrices of kernel weights, and ξ_2 is the 3×1 vector having 1 in the 2nd entry and zeros in the other entries, and $\xi_{p,x}$ is the $(p+1) \times 1$ vector having 1 in the p th entry and zeros in the other entries.

C.4 Proof of Proposition 3.7.1

By the KKT-condition, any minimizer $\check{\beta}$ of (Equation 6.34) satisfies:

$$-\frac{1}{NM} \widehat{\mathbf{F}}^T (\widehat{\mathbf{u}}_t - \widehat{\mathbf{F}} \check{\beta}) + \lambda_N \check{\mathbf{z}} = 0, \text{ for } \check{\mathbf{z}} \in \partial \|\check{\beta}\|_1. \quad (\text{C.7})$$

Recall that $\Delta \mathbf{u}_t = \hat{\mathbf{u}}_t - \mathbf{u}_t$, $\Delta \mathbf{F} = \hat{\mathbf{F}} - \mathbf{F}$ denote the error terms. By using the ground-truth PDE $u_t = \mathbf{F}\beta^*$ and definitions of $\Delta \mathbf{u}_t$ and $\Delta \mathbf{F}$, we have $\hat{\mathbf{u}}_t = \hat{\mathbf{F}}\beta^* - \Delta \mathbf{F}\beta^* + \Delta \mathbf{u}_t$. Thus from (Equation C.7), we get

$$\hat{\mathbf{F}}^T \hat{\mathbf{F}}(\check{\beta} - \beta^*) + \hat{\mathbf{F}}^T(\Delta \mathbf{F}\beta^* - \Delta \mathbf{u}_t) + \lambda_N NM \mathbf{z} = 0 . \quad (\text{C.8})$$

We decompose (Equation C.8) as follows:

$$\begin{bmatrix} \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S & \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_{S^c} \\ \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S & \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_{S^c} \end{bmatrix} \begin{bmatrix} \check{\beta}_S - \beta_S^* \\ 0 \end{bmatrix} + \begin{bmatrix} \hat{\mathbf{F}}_S^T \\ \hat{\mathbf{F}}_{S^c}^T \end{bmatrix} (\Delta \mathbf{F}_S \beta_S^* - \Delta \mathbf{u}_t) + \lambda_N NM \begin{bmatrix} \check{\mathbf{z}}_S \\ \check{\mathbf{z}}_{S^c} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} , \quad (\text{C.9})$$

where we used the fact $\beta_{S^c}^* = \mathbf{0}$ and $\check{\beta}_{S^c} = \mathbf{0}$ via PDW construction. Solving (Equation C.9), we have following two equalities:

$$\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S(\check{\beta}_S - \beta_S^*) + \hat{\mathbf{F}}_S^T(\Delta \mathbf{F}_S \beta_S^* - \Delta \mathbf{u}_t) + \lambda_N NM \check{\mathbf{z}}_S = 0 \quad (\text{C.10})$$

$$\hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S(\check{\beta}_S - \beta_S^*) + \hat{\mathbf{F}}_{S^c}^T(\Delta \mathbf{F}_S \beta_S^* - \Delta \mathbf{u}_t) + \lambda_N NM \check{\mathbf{z}}_{S^c} = 0 \quad (\text{C.11})$$

Using the minimum eigen-value condition in the assumption (Equation A3), from (Equation C.10), we have

$$\check{\beta}_S - \beta_S^* = (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \left(\hat{\mathbf{F}}_S^T(\Delta \mathbf{u}_t - \Delta \mathbf{F}_S \beta_S^*) - \lambda_N NM \check{\mathbf{z}}_S \right). \quad (\text{C.12})$$

Plugging (Equation C.12) into (Equation C.11) gives:

$$\check{\mathbf{z}}_{S^c} = \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \mathbf{z}_S + \frac{1}{\lambda_N NM} \hat{\mathbf{F}}_{S^c}^T \Pi_{S^\perp} (\Delta \mathbf{u}_t - \Delta \mathbf{F}_S \beta_S^*) ,$$

where $\Pi_{S^\perp} = \mathbf{I} - \hat{\mathbf{F}}_S (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \hat{\mathbf{F}}_S^T$ is an orthogonal projection operator on the column space of $\hat{\mathbf{F}}_S$. By the complementary slackness condition, for $j \in S^c$, $|\check{\mathbf{z}}_j| < 1$ implies $\check{\beta}_j = \mathbf{0}$, which guarantees the proper support recovery. i.e., $\mathcal{S}(\check{\beta}) \subseteq \mathcal{S}(\beta^*)$. Now, we can

focus on proving that, as $N, M \rightarrow \infty$, for μ in (Equation A3), $\mathbb{P}[\max_{j \in \mathcal{S}^c} |\tilde{Z}_j| \geq \mu] \rightarrow 0$, for $\tilde{Z}_j = [\hat{\mathbf{F}}_{\mathcal{S}^c}]_j^T \mathbf{\Pi}_{\mathcal{S}^\perp} \frac{\Delta \mathbf{u}_t - \Delta \mathbf{F}_{\mathcal{S}} \beta_{\mathcal{S}}^*}{\lambda N M}$, $[\hat{\mathbf{F}}_{\mathcal{S}^c}]_j$ is the j -th column of $\hat{\mathbf{F}}_{\mathcal{S}^c}$. By the following lemma, we claim that to prove (1) of Proposition 3.7.1, it suffices to bound ℓ_∞ -norm of the PDE estimation error $\boldsymbol{\tau}$.

Lemma C.4.1. *For any $\varepsilon > 0$:*

$$\mathbb{P}\left[\max_{j \in \mathcal{S}^c} |\tilde{Z}_j| \geq \varepsilon\right] \leq \mathbb{P}\left[\|\boldsymbol{\tau}\|_\infty \geq \frac{\lambda \varepsilon}{\sqrt{K}}\right].$$

Proof.

$$\begin{aligned} \mathbb{P}\left[\left\|\hat{\mathbf{F}}_{\mathcal{S}^c}^T \mathbf{\Pi}_{\mathcal{S}^\perp} \frac{\boldsymbol{\tau}}{\lambda N M}\right\|_\infty \geq \varepsilon\right] &\leq \mathbb{P}\left[\left\|\hat{\mathbf{F}}^T \mathbf{\Pi}_{\mathcal{S}^\perp} \frac{\boldsymbol{\tau}}{\lambda N M}\right\|_2 \geq \varepsilon\right] \\ &\leq \mathbb{P}\left[\left\|\mathbf{\Pi}_{\mathcal{S}^\perp}(\hat{\mathbf{F}})\right\|_2 \left\|\frac{\boldsymbol{\tau}}{\lambda N M}\right\|_2 \geq \varepsilon\right] \\ &\leq \mathbb{P}\left[\left\|\hat{\mathbf{F}}\right\|_F \left\|\frac{\boldsymbol{\tau}}{\lambda N M}\right\|_2 \geq \varepsilon\right] \\ &\leq \mathbb{P}\left[\|\boldsymbol{\tau}\|_2 \geq \lambda \varepsilon \sqrt{\frac{N M}{K}}\right] \\ &\leq \mathbb{P}\left[\|\boldsymbol{\tau}\|_\infty \geq \frac{\lambda \varepsilon}{\sqrt{K}}\right]. \end{aligned}$$

In the second inequality, we use the definition of spectral norm of matrix, and in the third inequality, we use the fact $\|\mathbf{\Pi}_{\mathcal{S}^\perp}\|_2 = 1$. In the fourth inequality, the condition $\frac{1}{\sqrt{N M}} \max_{j=1, \dots, K} \|\hat{\mathbf{F}}_j\|_2 \leq 1$ is used, giving us $\|\hat{\mathbf{F}}\|_F \leq \sqrt{K N M}$. In the last inequality, we use $\|\boldsymbol{\tau}\|_2 \leq \sqrt{N M} \|\boldsymbol{\tau}\|_\infty$. \square

C.4.1 Sufficient conditions for bounding $\hat{\mathbf{u}}_t - \mathbf{u}_t$

Lemma C.4.2. *Let $\mathcal{K}_{\max}^* = \|\mathcal{K}^*\|_\infty$, B_N be an arbitrary increasing sequence $B_N \rightarrow \infty$ as $N \rightarrow \infty$, and $B'_N = B_N + \|u\|_{L^\infty(\Omega)}$. For any $i = 0, 1, \dots, M$ and arbitrary real r , there exist finite positive constants $A(X_i)$, $C^*(X_i)$, a_0 , b_0 , c_0 , and $d_0(X_i)$ which do not depend on*

the temporal sample size N , such that for any $\alpha > 1$ and

$$\begin{aligned} \varepsilon_N^*(X_i, r, \alpha) &> \\ \max \left\{ 3|C^*(X_i)|h_N^2, \frac{6\mathcal{K}_{\max}^*B'_N}{Nh_N^2}, 6\frac{A(X_i)(B'_N)^{-1}}{h_N}, \frac{6B'_N\mathcal{K}_{\max}^*(a_0 \ln N + r) \ln N}{h_N^2 N}, \right. \\ &\left. 12\sqrt{\alpha}d_0(X_i)\sqrt{\frac{\ln 1/h_N}{h_N^3 N}} \right\}, \end{aligned}$$

as long as N is sufficiently large, we have:

$$\mathbb{P} \left[\sup_{t \in [0, T]} |\Delta u_t(X_i, t)| > \varepsilon_N^*(X_i, r, \alpha) \right] < 2N \exp \left(-\frac{B_N^2}{2\sigma^2} \right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 h_N^\alpha.$$

Proof. In the following argument, we fix some $i = 0, \dots, M-1$ and omit the dependence on X_i in the notations. Let $B'_N = B_N + \|u\|_{L^\infty(\Omega)}$ with B_N being a sequence of increasing positive numbers such that $B_N \rightarrow \infty$ as $N \rightarrow \infty$, then define the truncated estimate

$$\begin{aligned} \widehat{u}_t^{B'_N}(X_i, t) &= \frac{1}{Nh_N^2} \sum_{n=0}^{N-1} \mathcal{K}^* \left(\frac{t_n - t}{h_N} \right) U_i^n I\{|U_i^n| < B'_N\} \\ &= \frac{1}{h_N^2} \iint_{|y| < B'_N} \mathcal{K}^* \left(\frac{z - t}{h_N} \right) y df_N(z, y), \end{aligned} \quad (\text{C.13})$$

where $f_N(\cdot, \cdot) := f_N(\cdot, \cdot | X_i)$ is the empirical distribution of (t_n, U_i^n) conditioned on the space X_i . For any (X_i, t) , decomposing the estimation error of the temporal partial derivative as follows

$$\widehat{u}_t - u_t = \underbrace{\left(\widehat{u}_t - \widehat{u}_t^{B'_N} - \mathbb{E}(\widehat{u}_t - \widehat{u}_t^{B'_N}) \right)}_{\text{Asymptotic deviation on the truncation error}} + \underbrace{\left(\widehat{u}_t^{B'_N} - \mathbb{E}\widehat{u}_t^{B'_N} \right)}_{\text{Asymptotic deviation of truncated estimator}} + \underbrace{\left(\mathbb{E}\widehat{u}_t - u_t \right)}_{\text{Asymptotic bias}},$$

we will prove that the error is bounded (in probability) by showing each component is bounded.

Component 1. Asymptotic deviation on the truncation error: Notice that for any $\varepsilon_{0,N} \geq \frac{\mathcal{K}_{\max}^* B'_N}{N h_N^2}$:

$$\begin{aligned} \mathbb{P}\left[\sup_t |\hat{u}_t - \hat{u}_t^{B'_N}| > \varepsilon_{0,N}\right] &= \mathbb{P}\left[\sup_t \left|\frac{1}{N h_N^2} \sum_{n=0}^{N-1} \mathcal{K}^*\left(\frac{t_n - t}{h_N}\right) U_i^n I\{|U_i^n| \geq B'_N\}\right| > \varepsilon_{0,N}\right] \\ &\leq \mathbb{P}\left[\frac{\mathcal{K}_{\max}^*}{N h_N^2} \sum_{n=0}^{N-1} |U_i^n| I\{|U_i^n| \geq B'_N\} > \varepsilon_{0,N}\right] \leq \mathbb{P}\left[\exists n = 0, 1, \dots, N-1, |U_i^n| \geq B'_N\right] \\ &= \mathbb{P}\left[\max_{n=0,1,\dots,N-1} |U_i^n| \geq B'_N\right] \leq \mathbb{P}\left[\max_{n=0,1,\dots,N-1} |U_i^n - u_i^n| \geq B_N\right] \leq 2N \exp\left(-\frac{B_N^2}{2\sigma^2}\right) \end{aligned}$$

where σ denotes the standard deviation of the Gaussian noise added on the data. On the other hand, from Proposition 1 of [415]:

$$\mathbb{E} |\hat{u}_t - \hat{u}_t^{B'_N}| \leq \frac{A(B'_N)^{-1}}{h_N}.$$

for $A = \int |\mathcal{K}(\zeta)| d\zeta \times \sup_t \int |y| f(t, y|X_i) dy$ with $f(\cdot, \cdot|X_i)$ as the distribution of $(t, U(X_i, t))$; hence for any $\varepsilon_{1,N} \geq 2 \max\left\{\frac{\mathcal{K}_{\max}^* B_N}{N h_N^2}, \frac{A(B'_N)^{-1}}{h_N}\right\}$, we have:

$$\mathbb{P}\left[\sup_t |\hat{u}_t(X_i, t) - \hat{u}_t^{B'_N}(X_i, t) - (\mathbb{E}(\hat{u}_t(X_i, t) - \hat{u}_t^{B'_N}(X_i, t)))| > \varepsilon_{1,N}\right] \leq 2N \exp\left(-\frac{B_N^2}{2\sigma^2}\right).$$

Component 2. Asymptotic deviation of truncated estimator: Observe that

$$\begin{aligned} \hat{u}_t^{B'_N} - \mathbb{E}(\hat{u}_t^{B'_N}) &= \frac{1}{\sqrt{N} h_N^2} \int_{z \in \mathbb{R}} \int_{|y| \leq B'_N} \mathcal{K}^*\left(\frac{z-t}{h_N}\right) y d_z d_y \underbrace{\left(\sqrt{N}(f_N(z, y) - f(z, y))\right)}_{:= Z_N(z, y)} \\ &= \frac{1}{\sqrt{N} h_N^2} \int_{z \in \mathbb{R}} \mathcal{K}^*\left(\frac{z-t}{h_N}\right) d_z U^{B'_N}(z), \end{aligned} \tag{C.14}$$

where $U^{B'_N}(z)$ is defined by

$$U^{B'_N}(z) := \int_{|y| \leq B'_N} y d_y Z_N(z, y).$$

Let $\mathcal{T} : \mathbb{R}^2 \rightarrow [0, 1]^2$ be the Rosenblatt transformation [459], and define \mathcal{B} as the 2-dimensional solution path of the Brownian Bridge which takes the transformed $\mathcal{T}(z, y)$ as an argument; then we have

$$U^{B'_N}(z) := \int_{|y| \leq B'_N} y d_y \{Z_N(z, y) - \mathcal{B}(\mathcal{T}(z, y))\} + \int_{|y| \leq B'_N} y d_y \mathcal{B}(\mathcal{T}(z, y)). \quad (\text{C.15})$$

Plug in (Equation C.15) to (Equation C.14), we get

$$\begin{aligned} \widehat{u}_t^{B'_N} - \mathbb{E}(\widehat{u}_t^{B'_N}) &= \underbrace{\frac{1}{\sqrt{N}h_N^2} \int_{z \in \mathbb{R}} \mathcal{K}^*\left(\frac{z-t}{h_N}\right) d_z \int_{|y| \leq B'_N} y d_y \{Z_N(z, y) - \mathcal{B}(\mathcal{T}(z, y))\}}_{\gamma_N(t)} \\ &\quad + \underbrace{\frac{1}{\sqrt{N}} \frac{1}{h_N^2} \int_{z \in \mathbb{R}} \int_{|y| \leq B'_N} \mathcal{K}^*\left(\frac{z-t}{h_N}\right) y d_z d_y \mathcal{B}(\mathcal{T}(z, y))}_{\rho_N(t)} \\ &= \gamma_N(t) + \frac{1}{\sqrt{N}} \rho_N(t). \end{aligned}$$

In the following, we bound γ_N and $\rho_N(t)/\sqrt{N}$ respectively.

1. *Bound for $\gamma_N(t)$:* Since \mathcal{K}^* has compact support, applying integration by parts on $\gamma_N(t)$ gives

$$\begin{aligned} \gamma_N(t) &= -\frac{1}{\sqrt{N}h_N^2} \int_{z \in \mathbb{R}} \int_{|y| \leq B'_N} y d_y \{Z_N(z, y) - \mathcal{B}(\mathcal{T}(z, y))\} d_z \mathcal{K}^*\left(\frac{z-t}{h_N}\right) \\ &\leq \frac{2B'_N \mathcal{K}_{\max}^*}{\sqrt{N}h_N^2} \sup_{z, y} \left| Z_N(z, y) - \mathcal{B}(\mathcal{T}(z, y)) \right|. \end{aligned} \quad (\text{C.16})$$

By Tusnady's strong approximation result [416], there exist absolute positive constants a_0, b_0 and c_0 such that

$$\mathbb{P} \left[\sup_{z, y} \left| Z_N(z, y) - \mathcal{B}(\mathcal{T}(z, y)) \right| > \frac{(a_0 \ln N + r) \ln N}{\sqrt{N}} \right] < b_0 \exp(-c_0 r) \quad (\text{C.17})$$

holds for any real r . Therefore, if we take $\varepsilon'_{2,N}(r) = \frac{2B'_N \mathcal{K}_{\max}^* (a_0 \ln N + r) \ln N}{N h_N^2}$, combin-

ing (Equation C.16) and (Equation C.17) gives

$$\mathbb{P} \left[\sup_t |\gamma_N(t)| > \varepsilon'_{2,N}(r) \right] < b_0 \exp(-c_0 r). \quad (\text{C.18})$$

2. *Bound for $\rho_N(t)/\sqrt{N}$:* Similarly to (7) of [415], we have

$$\begin{aligned} \frac{h_N^{3/2} \sup_t |\rho_N(t)|}{\sqrt{\ln \frac{1}{h_N}}} &\leq \underbrace{16(\ln V)^{1/2} S^{1/2} \left(\ln \frac{1}{h_N} \right)^{-1/2} \int |\zeta|^{1/2} |d\mathcal{K}^*(\zeta)|}_{:=Q_{1,N}} \\ &\quad + \underbrace{16\sqrt{2}h_N^{-1/2} \left(\ln \frac{1}{h_N} \right)^{-1/2} \int q(Sh_N|\zeta|) |d\mathcal{K}^*(\zeta)|}_{:=Q_{2,N}}, \end{aligned}$$

where V is a random variable satisfying $\mathbb{E} V \leq 4\sqrt{2}\eta^4$ (recall that $\eta^2 := \max_{i,n} \mathbb{E}(U_i^n)^2$), $q(r) := \int_0^r \frac{1}{2} \left(\frac{1}{y} \ln \frac{1}{y} \right)^{1/2} dy$, $S := \sup_z \int y^2 f(z, y) dy$. Let $d_0 = 16\sqrt{2}S^{1/2} \int |\zeta|^{1/2} |d\mathcal{K}^*(\zeta)|$, which is a positive number independent of either N or M . Consider the following inequality for an arbitrary ε

$$\begin{aligned} \mathbb{P} \left(\frac{h_N^{3/2} \sup_t |\rho_N(t)|}{\sqrt{\ln \frac{1}{h_N}}} \geq \varepsilon \right) &\leq \mathbb{P} \left(Q_{1,N} \geq \frac{\varepsilon}{2} \right) + \mathbb{P} \left(Q_{2,N} \geq \frac{\varepsilon}{2} \right) \\ &\leq \mathbb{P} \left((\ln V)^{1/2} \geq \frac{\varepsilon \left(\ln \frac{1}{h_N} \right)^{1/2}}{2d_0} \right) + \mathbb{P} \left(Q_{2,N} \geq \frac{\varepsilon}{2} \right) \\ &\leq 4\sqrt{2}\eta^4 \exp \left(-\frac{\varepsilon^2 \left(\ln \frac{1}{h_N} \right)}{4d_0^2} \right) + \mathbb{P} \left(Q_{2,N} \geq \frac{\varepsilon}{2} \right), \end{aligned} \quad (\text{C.19})$$

where the Markov Inequality is used in the last inequality. Setting $\varepsilon''_{2,N} = \varepsilon \sqrt{\frac{\ln \frac{1}{h_N}}{N h_N^3}}$ gives

$$\mathbb{P} \left(\frac{\sup_t |\rho_N(t)|}{\sqrt{N}} \geq \varepsilon''_{2,N} \right) \leq 4\sqrt{2}\eta^4 \exp \left(-\frac{\varepsilon^2 \left(\ln \frac{1}{h_N} \right)}{4d_0^2} \right) + \mathbb{P} \left(Q_{2,N} \geq \frac{\varepsilon}{2} \right).$$

Notice that $Q_{2,N}$ converges to d_0 by Silverman [420]. For any arbitrary $\alpha > 1$, if

$\varepsilon = 2\sqrt{\alpha}d_0$, there exists a positive integer $N(\alpha)$ such that as long as $N > N(\alpha)$, we have $Q_{2,N} < \sqrt{\alpha}d_0$; hence the second probability in (Equation C.19) becomes 0. Considering that $\varepsilon''_{2,N}$ now depends on α , we write it as $\varepsilon''_{2,N}(\alpha)$, and for sufficiently large N ($N > N(\alpha)$), we obtain

$$\mathbb{P}\left(\frac{\sup_t |\rho_N(t)|}{\sqrt{N}} \geq \varepsilon''_{2,N}(\alpha)\right) \leq 4\sqrt{2}\eta^4 h_N^\alpha. \quad (\text{C.20})$$

Now if we take $\varepsilon_{2,N}(r, \alpha) = 2 \max\{\varepsilon'_{2,N}(r), \varepsilon''_{2,N}(\alpha)\}$ and combine (Equation C.18) with (Equation C.20), we have

$$\mathbb{P}\left(\sup_t |\hat{u}_t^{B'_N} - \mathbb{E}(\hat{u}_t^{B'_N})| > \varepsilon_{2,N}(r, \alpha)\right) < b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 h_N^\alpha$$

Component 3. Asymptotic bias: From [413], the asymptotic bias of the estimator directly follows

$$\mathbb{E}(\hat{u}_t) - u_t = C^* h_N^2.$$

for some constant C^* independent of N . Specifically, since we fit a degree 2 polynomial to obtain $\hat{u}_t(X_i, \cdot)$, we plug $p = 2$ and $\nu = 1$ in the expression of asymptotic bias of the estimator. See page 83 of the paper [413] for the expression. Taking $\varepsilon_{3,N} = |C^*| h_N^2$, we have $\mathbb{P}(|\mathbb{E}(\hat{u}_t) - u_t| > \varepsilon_{3,N}) = 0$.

Combining all the three components above and taking $\varepsilon_N^*(r, \alpha) > 3 \max\{\varepsilon_{1,N}, \varepsilon_{2,N}(r, \alpha), \varepsilon_{3,N}\}$ gives the desired result. \square

C.4.2 Sufficient conditions for bounding $(\hat{\mathbf{F}} - \mathbf{F})\beta^*$

For the p -th order partial derivative estimators with respect to x , we have results similarly to Lemma C.4.2.

Lemma C.4.3. Fix an order $p \geq 0$, and let B_M be an arbitrary increasing sequence $B_M \rightarrow \infty$ as $M \rightarrow \infty$, and $B'_M = B_M + \|u\|_{L^\infty(\Omega)}$. For any $n = 0, 1, \dots, N-1$ and arbitrary r , there exist finite positive constants $A_p(t_n)$, $C^*(t_n)$, a_0 , b_0 , c_0 , and $d_0(t_n)$ which do not depend on the spacial sample size M , such that for any $\alpha > 1$ and

$$\begin{aligned} \varepsilon_{M,p}^*(t_n, r, \alpha) &> \\ \max \left\{ 3|C^*(t_n)|w_M^2, \frac{6p!\mathcal{K}_{\max}^*B'_M}{Mw_M^{1+p}}, \right. \\ &\left. 6\frac{p!A_p(t_n)(B'_M)^{-1}}{w_M^p}, \frac{6p!B'_M(a_0 \ln M + r) \ln M}{w_M^{1+p}M}, 12p!\sqrt{\alpha}d_0(t_n)\sqrt{\frac{\ln 1/w_M}{w_M^{2p+1}M}} \right\}, \end{aligned}$$

as long as $M > M(\alpha)$ for some positive integer $M(\alpha)$, we have:

$$\mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\widehat{\partial_x^p u}(x, t_n) - \partial_x^p u(x, t_n)| > \varepsilon_{M,p}^* \right] < 2M \exp \left(-\frac{B_M^2}{2\sigma^2} \right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 w_M^\alpha.$$

Proof. Notice that for any fixed temporal point t_n , $n = 0, 1, \dots, N-1$, the estimation for the p -th order partial derivative takes the form

$$\widehat{\partial_x^p u}(x, t_n) = \frac{p!}{Mw_M^{p+1}} \sum_{i=1}^M \mathcal{K}^* \left(\frac{X_i - x}{w_M} \right) U_i^n \quad (\text{C.21})$$

with probability 1 [414]. Hence, we can prove the desired result by substituting h_N^2 with $w_M^{p+1}/p!$ in (Equation C.13) and follow the proof of Lemma C.4.2 and keeping in mind that the constants now depend on t_n and not on M . Notice that the kernel \mathcal{K} used for the spacial dimension may be different from that used for the temporal; this can be addressed by taking \mathcal{K}_{\max}^* to be the larger value between their ℓ_∞ -norms. Finally, given any fixed t_n , the asymptotic bias takes the form

$$\mathbb{E}(\widehat{\partial_x^p u}) - \partial_x^p u = C_p^* w_M^2$$

where $C_p^* \leq \max_{p=0,1,\dots,P_{\max}} \left\{ \int z^{p+1} \mathcal{K}_p^*(z) dz \right\} \frac{p!}{(p+2)!} \partial_x^{p+1} u := C^*$ for any $0 \leq p \leq P_{\max}$.

Here, since we fit the Local-Polynomial with degree $\ell + 1$ to obtain $\widehat{\partial_x^\ell u}$, we plug $p = \ell + 1$ and $\nu = \ell$ in the expression of asymptotic bias in [413]. \square

As for the product terms:

Lemma C.4.4. *Fix any two orders $p, q \geq 0$, and let B_M be an arbitrary increasing sequence $B_M \rightarrow \infty$ as $M \rightarrow \infty$, and $B'_M = B_M + \|u\|_{L^\infty(\Omega)}$. For any $n = 0, 1, \dots, N - 1$ and arbitrary r , there exist finite positive constants $A(t_n), C^*(t_n), a_0, b_0, c_0$, and $d_0(t_n)$ which do not depend on the spacial sample size M , such that for any $\alpha > 1$ and*

$$\varepsilon_{M,p,q}^{**} > \max\{3\|\partial_x^p u(\cdot, t_n)\|_\infty \varepsilon_{M,p}^*, 3\|\partial_x^q u(\cdot, t_n)\|_\infty \varepsilon_{M,q}^*, 3(\varepsilon_{M,p}^*)^2, 3(\varepsilon_{M,q}^*)^2\}$$

as long as $M > M(\alpha)$ for some positive integer $M(\alpha)$, we have:

$$\begin{aligned} & \frac{1}{4} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n)| > \varepsilon_{M,p,q}^{**} \right] \\ & < 2M \exp\left(-\frac{B_M^2}{2\sigma^2}\right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 w_M^\alpha, \end{aligned}$$

Here $\varepsilon_{M,p}^*$ and $\varepsilon_{M,q}^*$ (depending on B'_M) are the thresholds in Lemma C.4.3 for the sup-norm bound of the estimator $\widehat{\partial_x^p u}$ and $\widehat{\partial_x^q u}$, respectively,

Proof. Notice that for any $\varepsilon > 0$, we can bound the probability:

$$\begin{aligned} & \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n)| > \varepsilon \right] \\ & \leq \mathbb{P} \left[\|\partial_x^p u(\cdot, t_n)\|_\infty \sup_{x \in [0, X_{\max})} |\Delta \partial_x^q u(x, t_n)| > \varepsilon/3 \right] \\ & + \mathbb{P} \left[\|\partial_x^q u(\cdot, t_n)\|_\infty \sup_{x \in [0, X_{\max})} |\Delta \partial_x^p u(x, t_n)| > \varepsilon/3 \right] \\ & + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\Delta \partial_x^p u(x, t_n)| > \sqrt{\frac{\varepsilon}{3}} \right] + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\Delta \partial_x^q u(x, t_n)| > \sqrt{\frac{\varepsilon}{3}} \right], \end{aligned}$$

hence the results follow from Lemma C.4.3. \square

As for higher degree terms, we can take the similar approach to obtain general results but with more complicated notations. In this work, we focus on demonstrating the essence without involving more indices.

C.4.3 Simplification on the Probability Bounds

Before proceeding further, we simplify the expressions for ε_N^* as well as the probability bounds in Lemma C.4.2 by considering the window width h_N and the diverging sequence B_N as follows

$$h_N = \frac{1}{N^a}, \quad B_N = N^b.$$

Here $a, b > 0$ are positive coefficients to be determined.

Consequently, we update the expressions of the five terms whose maximum defines the threshold ε_N^*

$$\begin{aligned} E_1(N) &= \frac{3|C^*(X_i)|}{N^{2a}}, \quad E_2(N) = \frac{6\mathcal{K}_{\max}^*(N^b + \|u\|_{L^\infty(\Omega)})}{N^{1-2a}}, \quad E_3(N) = \frac{6A(X_i)}{N^{-a}(N^b + \|u\|_{L^\infty(\Omega)})} \\ E_4(N) &= \frac{6\mathcal{K}_{\max}^*(N^b + \|u\|_{L^\infty(\Omega)})(a_0 \ln N + r) \ln N}{N^{1-2a}}, \quad E_5(N) = 12\sqrt{\alpha}d_0(X_i)\sqrt{\frac{a \ln N}{N^{1-3a}}}. \end{aligned}$$

When N is sufficiently large, to determine ε_N^* , we only need to focus on comparing the powers of N in $E_i(N)$, $i = 1, 2, \dots, 5$; this immediately leads to:

$$E_2(N) = \mathcal{O}(E_4(N)),$$

hence it's sufficient to only consider $E_1(N)$, $E_2(N)$, $E_4(N)$, and $E_5(N)$. The optimal

choice of a and b is determined by requiring

$$\begin{cases} 2a = 1 - b - 2a \\ 2a = \frac{1-3a}{2} \end{cases} \implies \begin{cases} a = \frac{1}{7} \\ b = \frac{3}{7} \end{cases}$$

To summarize the discussion above, we have

Corollary C.4.1. *Let $h_N = N^{-1/7}$. For any $i = 0, 1, \dots, M$ and arbitrary real r , there exist finite positive constants $C^*(X_i)$, a_0 , b_0 , c_0 , and $d_0(X_i)$ which do not depend on the temporal sample size N , such that for N sufficiently large, any $\alpha > 1$, and*

$$\varepsilon_N^*(X_i, r, \alpha) > N^{-\frac{2}{7}} \max \left\{ 3|C^*(X_i)|, 6(a_0 \ln N + r) \ln N, 12\sqrt{\alpha}d_0(X_i)\sqrt{\frac{\ln N}{7}} \right\},$$

we have:

$$\mathbb{P} \left[\sup_{t \in [0, T]} |\Delta u_t(X_i, t)| > \varepsilon_N^*(X_i, r, \alpha) \right] < 2N \exp \left(-\frac{N^{6/7}}{2\sigma^2} \right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 N^{-\alpha/7},$$

Similarly, we can obtain optimal $w_M = M^{-1/(2p+5)}$ and $B_M = M^{(p+2)/(2p+5)}$ for the estimation of p -th partial derivative of u . Consequently, the threshold lower bound in Lemma C.4.3 becomes

$$\varepsilon_{M,p}^*(t_n, r, \alpha) > M^{-2/(2p+5)} \max \left\{ 3|C^*(t_n)|, 6p!(a_0 \ln M + r) \ln M, 12p!\sqrt{\alpha}d_0(t_n)\sqrt{\frac{\ln M}{2p+5}} \right\}.$$

Notice that the right hand side of the inequality above is non-decreasing with respect to $p \geq 0$. Moreover, note that for sufficiently large M , if the probability bound in Lemma C.4.3 holds for some w_M , then it holds for any smaller window width $w'_M < w_M$. Therefore, we have the following simplified result

Corollary C.4.2. *Let $w_M = M^{-1/7}$. For any $n = 0, 1, \dots, N-1$ and arbitrary r , there exist finite positive constants $C^*(t_n)$, a_0 , b_0 , c_0 , and $d_0(t_n)$ which do not depend on the*

spacial sample size M , such that for M sufficiently large, any $\alpha > 1$, and

$$\varepsilon_M^*(t_n, r, \alpha) > M^{-2/(2P_{\max}+5)} \max \left\{ 3|C^*(t_n)|, 6P_{\max}!(a_0 \ln M + r) \ln M, 12P_{\max}!\sqrt{\alpha}d_0(t_n)\sqrt{\frac{\ln M}{2P_{\max}+5}} \right\},$$

we have:

$$\begin{aligned} & \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\widehat{\partial_x^p u}(x, t_n) - \partial_x^p u(x, t_n)| > \varepsilon_M^* \right] \\ & < 2M \exp \left(- \frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2} \right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 M^{-\alpha/(2P_{\max}+5)} \end{aligned}$$

for any order $0 \leq p \leq P_{\max}$.

Similarly, for the product terms, we have

Corollary C.4.3. *Let $w_M = M^{-1/7}$. For any $n = 0, 1, \dots, N-1$ and arbitrary r , there exist finite positive constants $C^*(t_n)$, a_0 , b_0 , c_0 , and $d_0(t_n)$ which do not depend on the spacial sample size M , such that for M sufficiently large, any $\alpha > 1$, and*

$$\varepsilon_M^{**} > \max\{3\|u(\cdot, t_n)\|_{P_{\max}, \infty} \varepsilon_M^*, 3(\varepsilon_M^*)^2\}$$

where $\|u(\cdot, t_n)\|_{P_{\max}, \infty} = \sum_{0 \leq k \leq P_{\max}} \|\partial_x^k u(\cdot, t_n)\|_{\infty}$, we have

$$\begin{aligned} & \frac{1}{4} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} |\widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n)| > \varepsilon_M^{**} \right] \\ & < 2M \exp \left(- \frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2} \right) + b_0 \exp(-c_0 r) + 4\sqrt{2}\eta^4 M^{-\alpha/(2P_{\max}+5)} \end{aligned}$$

for any orders $0 \leq p, q \leq P_{\max}$.

C.4.4 ℓ_∞ Bound for the PDE Estimation Error τ

Notice that in the previous results, although the constants $C^*(X_i)$ and $d_0(X_i)$ are independent of N , they show dependence on the spacial point X_i . Similarly, $C^*(t_n)$ and $d_0(t_n)$ are independent of M , yet their values may depend on N . To guarantee that as both $N, M \rightarrow \infty$, these constants are uniformly bounded, we prove the following lemma.

Lemma C.4.5. *For any integer $M \geq 1$, and any $i = 0, 1, \dots, M-1$, $|C^*(X_i)|$ and $d_0(X_i)$ in Corollary C.4.1 are bounded by constants that are independent of M . That is, there exist constants $C^*, d_0 > 0$ such that for any $M \geq 1$*

$$\max_{i=0, \dots, M-1} |C^*(X_i)| \leq C^* \|\partial_t^3 u\|_\infty, \text{ and } \max_{i=0, \dots, M-1} d_0(X_i) \leq d_0.$$

Proof. From (3.7) in the Theorem 3.1 of [414], we have

$$|C^*(X_i)| \leq C^* \|\partial_t^3 u\|_\infty < \infty$$

where C^* only depends on the choice of the kernel function and the order of the Local-Polynomial. Recalling that $d_0(X_i) = 16S^{1/2} \int |\zeta|^{1/2} |d\mathcal{K}^*(\zeta)|$ where $S = \sup_z \int y^2 f(z, y|X_i) dy$. For a general real number s , we know that

$$\begin{aligned} \sup_{z \in [0, T_{\max}]} \int |y|^s f(z, y|X_i) dy &= \sup_{z \in [0, T_{\max}]} \int |y|^s \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left(-\frac{(y - u(X_i, z))^2}{2\sigma^2}\right) dy \\ &= \sup_{z \in [0, T_{\max}]} \sigma^s 2^{s/2} \frac{\Gamma(\frac{1+s}{2})}{\sqrt{\pi}} {}_1F_1\left(-\frac{s}{2}, \frac{1}{2}, -\frac{1}{2}\left(\frac{u(X_i, z)}{\sigma}\right)^2\right) \end{aligned}$$

where ${}_1F_1(p, q, w)$ is Kummer's confluent hyper-geometric function of $w \in \mathbb{C}$ with parameters $p, q \in \mathbb{C}$ (See, e.g.[460]) and Γ is the Gamma function. Since ${}_1F_1(-\frac{s}{2}, \frac{1}{2}, \cdot)$ is an

entire function for fixed parameters,

$$\sup_{z \in [0, T_{\max}]} \int |y|^s f(z, y | X_i) dy \leq \sup_{z \in [0, T_{\max}]} \sigma^s 2^{s/2} \frac{\Gamma\left(\frac{1+s}{2}\right)}{\sqrt{\pi}} \sup_{w \in \left[-\frac{\max_{x \in \Omega} u^2(x, z)}{2\sigma^2}, -\frac{\min_{x \in \Omega} u^2(x, z)}{2\sigma^2}\right]} {}_1F_1\left(-\frac{s}{2}, \frac{1}{2}, w\right) < \infty$$

which clearly does not depend on M . Taking $s = 2$, we can obtain that $d_0(X_i) \leq d_0$ for some d_0 that only depends on the choice of kernel \mathcal{K} , underlying function $\|u\|_{L^\infty(\Omega)}$, and noise level σ . \square

Note that the same proof can derive that the constants in Lemma C.4.3 and Lemma C.4.4 are also bounded by N -independent constants. This technical lemma allows us to state

Proposition C.4.1. *Take $h_N = N^{-1/7}$ in the temporal direction and $w_M = M^{-1/7}$ in the space direction. There exist constants C , a_0 , b_0 , and c_0 which do not depend on N nor M such that for N and M sufficiently large, any r , $\alpha > 1$, and*

$$\varepsilon_{N,M}(r, \alpha) > C \max \left\{ \frac{(a_0 \ln N + r) \ln N}{N^{2/7}}, \frac{\sqrt{\alpha \ln N}}{N^{2/7}}, \frac{(a_0 \ln M + r) \ln M}{M^{2/(2P_{\max}+5)}}, \sqrt{\frac{\alpha \ln M}{(2P_{\max} + 5)M^{4/(2P_{\max}+5)}}} \right\}$$

we have

$$\begin{aligned} & \mathbb{P} \left[\|\boldsymbol{\tau}\|_\infty > \varepsilon_{N,M} \right] < \\ & 2NM \exp \left(-\frac{N^{6/7}}{2\sigma^2} \right) + b_0 \exp(-c_0 r) M + 4\sqrt{2}\eta^4 M N^{-\alpha/7} + \\ & 8sNM \exp \left(-\frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2} \right) + 4sb_0 \exp(-c_0 r) N + 16\sqrt{2}\eta^4 sNM^{-\alpha/(2P_{\max}+5)} \end{aligned}$$

Here K is the number of feature variables in the dictionary.

Proof. By triangle inequality, the ℓ_∞ -norm of PDE estimation error $\boldsymbol{\tau}$ (Equation 6.33) can be bounded by

$$\|\boldsymbol{\tau}\|_\infty \leq \|\Delta \mathbf{F} \boldsymbol{\beta}^*\|_\infty + \|\Delta \mathbf{u}_t\|_\infty.$$

By Corollary C.4.1 and Lemma C.4.5, there exists a constant C_1 independent of N and M such that with sufficiently large N and any $\varepsilon_N(r, \alpha) > C_1 N^{-2/7} \max\{(a_0 \ln N + r) \ln N, \sqrt{\alpha \ln N}\}$, we have

$$\begin{aligned} \mathbb{P}\left[\|\Delta \mathbf{u}_t\|_\infty > \varepsilon_N(r, \alpha)\right] &\leq \mathbb{P}\left[\max_{i=0,1,\dots,M-1} \sup_{t \in [0, T_{\max}]} |\Delta u_t(X_i, t)| > \varepsilon_N(r, \alpha)\right] \\ &\leq \sum_{i=0}^{M-1} \mathbb{P}\left[\sup_{t \in [0, T_{\max}]} |\Delta u_t(X_i, t)| > \varepsilon_N(r, \alpha)\right] \\ &< 2NM \exp\left(-\frac{N^{6/7}}{2\sigma^2}\right) + b_0 \exp(-c_0 r)M + 4\sqrt{2}\eta^4 MN^{-\alpha/7}. \end{aligned}$$

On the other hand, if we denote $\Delta F_k(x, t)$ as the approximation error of the k -th feature variable at time t and space x , we have

$$\|\Delta \mathbf{F} \boldsymbol{\beta}^*\|_\infty \leq \max_{n=0,1,\dots,N} \|\boldsymbol{\beta}^*\|_\infty \sup_{x \in [0, X_{\max})} \sum_{k=1}^s |\Delta F_k(x, t_n)|.$$

By Corollary C.4.2 and C.4.3, there exists a constant C_2 independent of N and M such that with sufficiently large M and any $\varepsilon_{K,M}(r, \alpha) > C_2 P_{\max}! K \|\boldsymbol{\beta}^*\|_\infty M^{-2/(2P_{\max}+5)} \max\{(a_0 \ln M + r) \ln M, \sqrt{\frac{\alpha \ln M}{2P_{\max}+5}}\}$, we have

$$\begin{aligned} \mathbb{P}\left[\|\Delta \mathbf{F} \boldsymbol{\beta}^*\|_\infty > \varepsilon_M(r, \alpha)\right] &\leq \sum_{n=0}^{N-1} \sum_{k=1}^s \mathbb{P}\left[\sup_{x \in [0, X_{\max})} |\Delta F_k(x, t_n)| > \frac{\varepsilon_M(r, \alpha)}{s \|\boldsymbol{\beta}^*\|_\infty}\right] \\ &< 8NMs \exp\left(-\frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2}\right) + 4b_0 \exp(-c_0 r)Ns + 16\sqrt{2}\eta^4 NsM^{-\alpha/(2P_{\max}+5)}. \end{aligned}$$

Taking $C = \max\{2C_1, 2s\|\boldsymbol{\beta}^*\|_\infty C_2 P_{\max}!\}$ proves the theorem. \square

C.4.5 Further Simplification

We further simplify our result by taking $M = N^b$ for some coefficient $b > 0$. Since r and α are arbitrary, we can vary them as we increase M, N by taking $r = N^c$ and $\alpha = N^d$ for some positive coefficients $c > 0$ and $d > 0$, respectively. Consequently, we have the lower

bound for $\varepsilon_{N,M}$ in Proposition C.4.1 becoming

$$\varepsilon_{N,M}(r, \alpha) > C \max \left\{ \frac{(a_0 \ln N + N^c) \ln N}{N^{2/7}}, \frac{\sqrt{\ln N}}{N^{2/7-d/2}}, \frac{b(a_0 b \ln N + N^c) \ln N}{N^{2b/(2P_{\max}+5)}}, \sqrt{\frac{b \ln N}{(2P_{\max} + 5)N^{4b/(2P_{\max}+5)-d}}} \right\}, \quad (\text{C.22})$$

To guarantee that the lower bound (Equation C.22) converges to 0 as $N \rightarrow \infty$, we have the following constraints on positive coefficients b, c , and d

$$\begin{cases} 0 < c < 2/7 \\ 2/7 - d/2 > 0 \\ c < 2b/(2P_{\max} + 5) \\ 4b/(2P_{\max} + 5) - d > 0 \end{cases}$$

Furthermore, we take $d = 2c$ so that

$$\frac{\sqrt{\ln N}}{N^{2/7-d/2}} = \mathcal{O} \left(\frac{(a_0 \ln N + N^c) \ln N}{N^{2/7}} \right), \quad \sqrt{\frac{b \ln N}{N^{4b/(2P_{\max}+5)-d}}} = \mathcal{O} \left(\frac{b(a_0 b \ln N + N^c) \ln N}{N^{2b/(2P_{\max}+5)}} \right).$$

and we can focus on the second and fourth term in (Equation C.22). As a result, the optimal choice for b is computed by $2/7 = 2b/(2P_{\max} + 5) \implies b = (2P_{\max} + 5)/7$. Based on the set-ups above, we obtain that for N sufficiently large, with

$$\varepsilon_N(c) > C \frac{\ln N}{N^{2/7-c}}$$

for any $0 < c < 2/7$, we have

$$\begin{aligned}
\mathbb{P}\left[\|\boldsymbol{\tau}\|_\infty > \varepsilon_N(c)\right] &< \\
&2N^{(2P_{\max}+12)/7} \exp\left(-\frac{N^{6/7}}{2\sigma^2}\right) + b_0 \exp(-c_0 N^c) N^{(2P_{\max}+5)/7} + 4\sqrt{2}\eta^4 N^{-N^{2c}/7} + \\
&8N^{(2P_{\max}+12)/7} K \exp\left(-\frac{N^{(2P_{\max}+5)/7}}{2\sigma^2}\right) + 4b_0 \exp(-c_0 N^c) NK + 16\sqrt{2}\eta^4 K N^{-N^{2c}/7} \\
&= \mathcal{O}\left(N^{\frac{2P_{\max}+5}{7}} \exp\left(-\frac{1}{6}N^c\right)\right),
\end{aligned}$$

where in the last equality, we plug $b_0 = 2$ and $c_0 = \frac{1}{6}$ from [461]. Combining this with Lemma C.4.1 proves the first part of the Proposition 3.7.1.

C.4.6 Proof of ℓ_∞ bound in (Equation 6.36)

Recall that in (Equation C.12), we have

$$\check{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^* = (\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} \left(\widehat{\mathbf{F}}_S^T (\Delta \mathbf{u}_t - \Delta \mathbf{F}_S \boldsymbol{\beta}_S^*) - \lambda_N NM \check{\mathbf{z}}_S \right).$$

Now, we are ready to bound the $\|\widehat{\boldsymbol{\beta}}_S^\lambda - \boldsymbol{\beta}_S^*\|_{\ell_\infty}$ bound in (Equation 6.36) as follows:

$$\begin{aligned}
\max_{k \in \mathcal{S}} |\beta_k - \beta_k^*| &\leq \left\| (\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} \right\|_2 \|\widehat{\mathbf{F}}_S^T \boldsymbol{\tau}\|_\infty + \lambda NM \left\| (\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} \right\|_2 \\
&\leq \left\| (\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S / (NM))^{-1} \right\|_2 \left(\|\widehat{\mathbf{F}}_S^T \boldsymbol{\tau}\|_\infty / (NM) + \lambda \right) \\
&\stackrel{(\text{item A1})}{\leq} \sqrt{K} C_{\min} \left(\|\widehat{\mathbf{F}}_S^T \boldsymbol{\tau}\|_\infty / (NM) + \lambda \right) \\
&\leq \sqrt{K} C_{\min} \left(\|\boldsymbol{\tau}\|_\infty \frac{\|\widehat{\mathbf{F}}_S\|_{\infty, \infty}}{NM} + \lambda \right) \\
&\leq \sqrt{K} C_{\min} \left(\|\boldsymbol{\tau}\|_\infty \frac{\|\widehat{\mathbf{F}}\|_F}{\sqrt{NM}} + \lambda \right) \\
&\leq \sqrt{K} C_{\min} \left(K \|\boldsymbol{\tau}\|_\infty + \lambda \right),
\end{aligned}$$

where we use normalized columns of $\widehat{\mathbf{F}}$ in the last inequality. Following the set-ups from Proposition 3.7.1 gives the desired result.

C.5 Proofs of Lemmas 6.8.1 and 6.8.2

Corollary C.5.1. *Fix any four orders $p, q, k \geq 0$, and let B_M be an arbitrary increasing sequence $B_M \rightarrow \infty$ as $M \rightarrow \infty$, and $B'_M = B_M + \|u\|_{L^\infty(\Omega)}$. For any $n = 0, 1, \dots, N-1$ and arbitrary r , there exist finite positive constants $A(t_n), C^*(t_n), a_0, b_0, c_0$, and $d_0(t_n)$ which do not depend on the spacial sample size M , such that for any $\alpha > 1$ and*

$$\varepsilon_{M,p,q,k}^{***} > \max \left\{ 3\|\partial_x^k u(\cdot, t_n)\|_\infty \varepsilon_{M,p,q}^{**}, 3\|\partial_x^p u(\cdot, t_n) \partial_x^q u(\cdot, t_n)\|_\infty \varepsilon_{M,k}^{**}, 3(\varepsilon_{M,p,q}^{**})^2, 3(\varepsilon_{M,k}^{**})^2 \right\}$$

as long as $M > M(\alpha)$ for some positive integer $M(\alpha)$, we have:

$$\begin{aligned} & \frac{1}{4} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) \widehat{\partial_x^k u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \partial_x^k u(x, t_n) \right| > \varepsilon_{M,p,q,k}^{***} \right] \\ & < 8M \exp \left(- \frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2} \right) + 4b_0 \exp(-c_0 r) + 16\sqrt{2}\eta^4 M^{-\alpha/(2P_{\max}+5)}, \end{aligned}$$

Here $\varepsilon_{M,p,q}^{**}$ and $\varepsilon_{M,k,l}^{**}$ (depending on B'_M) are the thresholds in Corollary C.4.3 for the sup-norm bound of the estimator $\widehat{\partial_x^p u} \widehat{\partial_x^q u}$ and $\widehat{\partial_x^k u} \widehat{\partial_x^l u}$, respectively,

Proof. Notice that for any $\varepsilon > 0$, we can bound the probability:

$$\begin{aligned}
& \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) \widehat{\partial_x^k u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \partial_x^k u(x, t_n) \right| > \varepsilon \right] \\
& \leq \mathbb{P} \left[\left\| \partial_x^k u(\cdot, t_n) \right\|_{\infty} \sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \right| > \varepsilon/3 \right] \\
& + \mathbb{P} \left[\left\| \partial_x^p u(\cdot, t_n) \partial_x^q u(\cdot, t_n) \right\|_{\infty} \sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^k u}(x, t_n) - \partial_x^k u(x, t_n) \right| > \varepsilon/3 \right] \\
& + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \right| > \sqrt{\frac{\varepsilon}{3}} \right] \\
& + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^k u}(x, t_n) - \partial_x^k u(x, t_n) \right| > \sqrt{\frac{\varepsilon}{3}} \right],
\end{aligned}$$

hence the results follow from corollary C.4.3. \square

Corollary C.5.2. Fix any four orders $p, q, k, l \geq 0$, and let B_M be an arbitrary increasing sequence $B_M \rightarrow \infty$ as $M \rightarrow \infty$, and $B'_M = B_M + \|u\|_{L^\infty(\Omega)}$. For any $n = 0, 1, \dots, N-1$ and arbitrary r , there exist finite positive constants $A(t_n), C^*(t_n), a_0, b_0, c_0$, and $d_0(t_n)$ which do not depend on the spacial sample size M , such that for any $\alpha > 1$ and

$$\begin{aligned}
& \varepsilon_{M,p,q,k,l}^{****} > \\
& \max \left\{ 3 \left\| \partial_x^p u(\cdot, t_n) \partial_x^q u(\cdot, t_n) \right\|_{\infty} \varepsilon_{M,p,q}^{**}, 3 \left\| \partial_x^k u(\cdot, t_n) \partial_x^l u(\cdot, t_n) \right\|_{\infty} \varepsilon_{M,k,l}^{**}, 3(\varepsilon_{M,p,q}^{**})^2, 3(\varepsilon_{M,k,l}^{**})^2 \right\}
\end{aligned}$$

as long as $M > M(\alpha)$ for some positive integer $M(\alpha)$, we have:

$$\begin{aligned}
& \frac{1}{4} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) \widehat{\partial_x^k u}(x, t_n) \widehat{\partial_x^l u}(x, t_n) \right. \right. \\
& \quad \left. \left. - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \partial_x^k u(x, t_n) \partial_x^l u(x, t_n) \right| > \varepsilon_{M,p,q,k,l}^{****} \right] \\
& < 8M \exp \left(- \frac{M^{(2P_{\max}+4)/(2P_{\max}+5)}}{2\sigma^2} \right) + 4b_0 \exp(-c_0 r) + 16\sqrt{2}\eta^4 M^{-\alpha/(2P_{\max}+5)},
\end{aligned}$$

Here $\varepsilon_{M,p,q}^{**}$ and $\varepsilon_{M,k,l}^{**}$ (depending on B'_M) are the thresholds in Corollary C.4.3 for the sup-norm bound of the estimator $\widehat{\partial_x^p u} \widehat{\partial_x^q u}$ and $\widehat{\partial_x^k u} \widehat{\partial_x^l u}$, respectively,

Proof. Notice that for any $\varepsilon > 0$, we can bound the probability:

$$\begin{aligned}
& \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) \widehat{\partial_x^k u}(x, t_n) \widehat{\partial_x^l u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \partial_x^k u(x, t_n) \partial_x^l u(x, t_n) \right| > \varepsilon \right] \\
& \leq \mathbb{P} \left[\left\| \partial_x^k u(\cdot, t_n) \partial_x^l u(\cdot, t_n) \right\|_\infty \sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \right| > \varepsilon/3 \right] \\
& + \mathbb{P} \left[\left\| \partial_x^p u(\cdot, t_n) \partial_x^q u(\cdot, t_n) \right\|_\infty \sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^k u}(x, t_n) \widehat{\partial_x^l u}(x, t_n) - \partial_x^k u(x, t_n) \partial_x^l u(x, t_n) \right| > \varepsilon/3 \right] \\
& + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^p u}(x, t_n) \widehat{\partial_x^q u}(x, t_n) - \partial_x^p u(x, t_n) \partial_x^q u(x, t_n) \right| > \sqrt{\frac{\varepsilon}{3}} \right] \\
& + \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\partial_x^k u}(x, t_n) \widehat{\partial_x^l u}(x, t_n) - \partial_x^k u(x, t_n) \partial_x^l u(x, t_n) \right| > \sqrt{\frac{\varepsilon}{3}} \right],
\end{aligned}$$

hence the results follow from corollary C.4.3. \square

Lemma C.5.1. *Let $\varepsilon_M^*, \varepsilon_M^{**}, \varepsilon_M^{***}, \varepsilon_M^{****}$ be the thresholds defined in corollaries C.4.2, C.4.3, C.5.1, and C.5.2. Then for any $\varepsilon_M^{\max'}$ such that*

$$\varepsilon_M^{\max'} > \sqrt{s(K-s)} \max \left\{ \varepsilon_M^*, \varepsilon_M^{**}, \varepsilon_M^{***}, \varepsilon_M^{****} \right\},$$

then, for $0 < c < \frac{2}{7}$, and for sufficiently large enough N , we have

$$\mathbb{P} \left[\frac{1}{NM} \left\| \widehat{\mathbf{F}}_{S^c}^T \widehat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_2 > \varepsilon_M^{\max'} \right] \leq \mathcal{O} \left(N \exp \left(-\frac{1}{6} N^c \right) \right).$$

Proof.

$$\begin{aligned}
& \mathbb{P} \left[\frac{1}{NM} \left\| \widehat{\mathbf{F}}_{\mathcal{S}^c}^T \widehat{\mathbf{F}}_{\mathcal{S}} - \mathbf{F}_{\mathcal{S}^c}^T \mathbf{F}_{\mathcal{S}} \right\|_2 > \varepsilon_M^{\max'} \right] \\
& \leq \mathbb{P} \left[\left\| \widehat{\mathbf{F}}_{\mathcal{S}^c}^T \widehat{\mathbf{F}}_{\mathcal{S}} - \mathbf{F}_{\mathcal{S}^c}^T \mathbf{F}_{\mathcal{S}} \right\|_{\mathbf{F}} > NM \varepsilon_M^{\max'} \right] \\
& \leq \mathbb{P} \left[\left\| \widehat{\mathbf{F}}_{\mathcal{S}^c}^T \widehat{\mathbf{F}}_{\mathcal{S}} - \mathbf{F}_{\mathcal{S}^c}^T \mathbf{F}_{\mathcal{S}} \right\|_{\infty, \infty} > NM \frac{\varepsilon_M^{\max'}}{\sqrt{s(K-s)}} \right] \\
& \leq \mathbb{P} \left[\max_{n=0, \dots, N-1} \sup_{x \in [0, X_{\max})} \left| \widehat{\mathbf{F}}_i(x, t_n) \widehat{\mathbf{F}}_j(x, t_n) - \mathbf{F}_i(x, t_n) \mathbf{F}_j(x, t_n) \right| > \frac{\varepsilon_M^{\max'}}{\sqrt{s(K-s)}} \right] \\
& \leq \sum_{n=0}^{N-1} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\mathbf{F}}_i(x, t_n) \widehat{\mathbf{F}}_j(x, t_n) - \mathbf{F}_i(x, t_n) \mathbf{F}_j(x, t_n) \right| > \frac{\varepsilon_M^{\max'}}{\sqrt{s(K-s)}} \right] \\
& \leq \mathcal{O} \left(N \exp \left(-\frac{1}{6} N^c \right) \right),
\end{aligned}$$

where we use the results from corollaries C.4.2, C.4.3, C.5.1, and C.5.2, and simplification argument used in the Appendix subsection C.4.5 in the last inequality. \square

C.5.1 Proof of Lemma 6.8.1

Proof. Observe that we can write:

$$\begin{aligned}
\Lambda_{\min} \left(\frac{1}{NM} \mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} \right) &:= \frac{1}{NM} \min_{\|x\|_2=1} x^T \left(\mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} \right) x \\
&= \frac{1}{NM} \min_{\|x\|_2=1} \left\{ x^T \left(\widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right) x + x^T \left(\mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} - \widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right) x \right\} \\
&\leq \frac{1}{NM} \left\{ y^T \left(\widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right) y + y^T \left(\mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} - \widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right) y \right\}
\end{aligned}$$

where $y \in \mathbb{R}^K$ is a unit-norm minimal eigen-vector of $\frac{1}{NM} \mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}}$. Therefore, we can write,

$$\begin{aligned}
\Lambda_{\min} \left(\frac{1}{NM} \widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right) &\geq \Lambda_{\min} \left(\frac{1}{NM} \mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} \right) - \frac{1}{NM} \left\| \mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} - \widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} \right\|_2 \\
&\geq C_{\min} - \frac{1}{NM} \left\| \widehat{\mathbf{F}}_{\mathcal{S}}^T \widehat{\mathbf{F}}_{\mathcal{S}} - \mathbf{F}_{\mathcal{S}}^T \mathbf{F}_{\mathcal{S}} \right\|_2.
\end{aligned}$$

By using a similar argument used in Lemma C.5.1, we can prove $\frac{1}{NM} \left\| \widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S - \mathbf{F}_S^T \mathbf{F}_S \right\|_2 \rightarrow 0$ with high-probability as $N \rightarrow \infty$. For any ε_M^{\max} such that,

$$\varepsilon_M^{\max} > s \max \left\{ \varepsilon_M^*, \varepsilon_M^{**}, \varepsilon_M^{***}, \varepsilon_M^{****} \right\},$$

Then, we can bound the probability as follows:

$$\begin{aligned} & \mathbb{P} \left[\frac{1}{NM} \left\| \widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S - \mathbf{F}_S^T \mathbf{F}_S \right\|_2 > \varepsilon_M^{\max} \right] \\ & \leq \mathbb{P} \left[\left\| \widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S - \mathbf{F}_S^T \mathbf{F}_S \right\|_F > NM \varepsilon_M^{\max} \right] \leq \mathbb{P} \left[\left\| \widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S - \mathbf{F}_S^T \mathbf{F}_S \right\|_{\infty, \infty} > NM \frac{\varepsilon_M^{\max}}{s} \right] \\ & \leq \mathbb{P} \left[\max_{n=0, \dots, N-1} \sup_{x \in [0, X_{\max})} \left| \widehat{\mathbf{F}}_i(x, t_n) \widehat{\mathbf{F}}_j(x, t_n) - \mathbf{F}_i(x, t_n) \mathbf{F}_j(x, t_n) \right| > \frac{\varepsilon_M^{\max}}{s} \right] \\ & \leq \sum_{n=0}^{N-1} \mathbb{P} \left[\sup_{x \in [0, X_{\max})} \left| \widehat{\mathbf{F}}_i(x, t_n) \widehat{\mathbf{F}}_j(x, t_n) - \mathbf{F}_i(x, t_n) \mathbf{F}_j(x, t_n) \right| > \frac{\varepsilon_M^{\max}}{s} \right] \\ & \leq \mathcal{O} \left(N \exp \left(-\frac{1}{6} N^c \right) \right). \end{aligned}$$

□

C.5.2 Proof of Lemma 6.8.2

Proof. Motivated from [411], we begin the proof by decomposing the sample matrix

$(\widehat{\mathbf{F}}_{S^c}^T \widehat{\mathbf{F}}_S)(\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1}$ into four parts:

$$\begin{aligned} (\widehat{\mathbf{F}}_{S^c}^T \widehat{\mathbf{F}}_S)(\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} &= \underbrace{\mathbf{F}_{S^c}^T \mathbf{F}_S \left((\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right)}_{:=\mathbf{T}_1} + \underbrace{\left(\widehat{\mathbf{F}}_{S^c}^T \widehat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right) (\mathbf{F}_S^T \mathbf{F}_S)^{-1}}_{:=\mathbf{T}_2} \\ &+ \underbrace{\left(\widehat{\mathbf{F}}_{S^c}^T \widehat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right) \left((\widehat{\mathbf{F}}_S^T \widehat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right)}_{:=\mathbf{T}_3} \\ &+ \underbrace{(\mathbf{F}_{S^c}^T \mathbf{F}_S) (\mathbf{F}_S^T \mathbf{F}_S)^{-1}}_{:=\mathbf{T}_4}. \end{aligned}$$

Since we know $\|\mathbf{T}_4\|_\infty \leq 1 - \mu$ for some $\mu \in (0, 1]$, the decomposition reduces the proof showing $\|\mathbf{T}_i\|_\infty \rightarrow 0$ with probability $1 - \mathcal{O}(N \exp(-\frac{1}{6}N^c))$ for $i = 1, 2, 3$.

1. Control of \mathbf{T}_1 : Observe that we can re-factorize \mathbf{T}_1 as follows:

$$\mathbf{T}_1 = (\mathbf{F}_{S^c}^T \mathbf{F}_S) (\mathbf{F}_S^T \mathbf{F}_S)^{-1} [\mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S] (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1}.$$

Then, by taking the advantage of sub-multiplicative property $\|AB\|_\infty \leq \|A\|_\infty \|B\|_\infty$ and the fact $\|\mathbf{T}_4\|_\infty \leq 1 - \mu$ and $\|C\|_\infty \leq \sqrt{N} \|C\|_2$ for $C \in \mathbb{R}^{M \times N}$, we can bound $\|\mathbf{T}_1\|_\infty$ as follows:

$$\begin{aligned} \|\mathbf{T}_1\|_\infty &\leq \left\| (\mathbf{F}_{S^c}^T \mathbf{F}_S) (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right\|_\infty \left\| \mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S \right\|_\infty \left\| (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \right\|_\infty \\ &\leq s(1 - \mu) \left(\frac{1}{NM} \left\| \mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S \right\|_2 \right) \left(NM \left\| (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \right\|_2 \right) \\ &\leq \frac{s(1 - \mu)}{C_{\min}} \left(\frac{1}{NM} \left\| \mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S \right\|_2 \right). \end{aligned}$$

Note that we use $\left\| (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \right\|_2 \leq \frac{1}{NM C_{\min}}$ with probability $1 - \mathcal{O}(N \exp(-\frac{1}{6}N^c))$ in the last inequality from Lemma 6.8.1.

2. Control of \mathbf{T}_2 : With similar techniques employed for controlling $\|\mathbf{T}_1\|_\infty$, we can bound $\|\mathbf{T}_2\|_\infty$ as follows:

$$\begin{aligned} \|\mathbf{T}_2\|_\infty &\leq \left\| \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_\infty \left\| (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right\|_\infty \\ &\leq s \left\| \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_2 \left\| (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right\|_2 \\ &= s \left(\frac{1}{NM} \left\| \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_2 \right) \left(NM \left\| (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} \right\|_2 \right) \\ &\leq \frac{s}{C_{\min}} \left(\frac{1}{NM} \left\| \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_2 \right). \end{aligned}$$

3. Control of \mathbf{T}_3 : To bound $\|\mathbf{T}_3\|_\infty$, we re-factorize the second argument of product in \mathbf{T}_3 :

$$(\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1} = (\mathbf{F}_S^T \mathbf{F}_S)^{-1} [(\mathbf{F}_S^T \mathbf{F}_S) - (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)] (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1}$$

With the factorization, we bound $\|(\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1}\|_\infty$ by using sub-multiplicative property and the fact $\|C\|_\infty \leq \sqrt{N}\|C\|_2$ for any $C \in \mathbb{R}^{M \times N}$ again:

$$\begin{aligned} \|(\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1}\|_\infty &= \|(\mathbf{F}_S^T \mathbf{F}_S)^{-1} [(\mathbf{F}_S^T \mathbf{F}_S) - (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)] (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1}\|_\infty \\ &\leq \sqrt{s} \|(\mathbf{F}_S^T \mathbf{F}_S)^{-1} [(\mathbf{F}_S^T \mathbf{F}_S) - (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)] (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1}\|_2 \\ &\leq \sqrt{s} \|(\mathbf{F}_S^T \mathbf{F}_S)^{-1}\|_2 \|[(\mathbf{F}_S^T \mathbf{F}_S) - (\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)]\|_2 \|(\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1}\|_2 \\ &\leq \frac{\sqrt{s}}{NMC_{\min}^2} \left(\frac{1}{NM} \|\mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S\|_2 \right). \end{aligned} \quad (\text{C.23})$$

Now we can bound $\|\mathbf{T}_3\|_\infty$ as follows:

$$\begin{aligned} \|\mathbf{T}_3\|_\infty &= \left\| \left(\hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right) \left((\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1} \right) \right\|_\infty \\ &\leq \left\| \hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S \right\|_\infty \|(\hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S)^{-1} - (\mathbf{F}_S^T \mathbf{F}_S)^{-1}\|_\infty \\ &\leq \frac{s}{C_{\min}} \left(\frac{1}{NM} \|\hat{\mathbf{F}}_{S^c}^T \hat{\mathbf{F}}_S - \mathbf{F}_{S^c}^T \mathbf{F}_S\|_2 \right) \left(\frac{1}{NM} \|\mathbf{F}_S^T \mathbf{F}_S - \hat{\mathbf{F}}_S^T \hat{\mathbf{F}}_S\|_2 \right), \end{aligned}$$

where in the last inequality, we use (Equation C.23) and $\|C\|_\infty \leq \sqrt{N}\|C\|_2$ for any $C \in \mathbb{R}^{M \times N}$. Take $\varepsilon_M^{\max''}$ such that, for $\varepsilon_M^{\max'}$ and ε_M^{\max} in Lemma C.5.1 and Lemma 6.8.1 respectively:

$$\varepsilon_M^{\max''} > \max \left\{ \frac{C_{\min}}{s(1-\mu)} \varepsilon_M^{\max}, \frac{C_{\min}}{s} \varepsilon_M^{\max'} \right\},$$

for large enough N , we have

$$\mathbb{P} \left[\forall i = 1, 2, 3 : \|\mathbf{T}_i\|_\infty > \varepsilon_M^{\max''} \right] \leq \mathcal{O} \left(N \exp \left(-\frac{1}{6} N^c \right) \right).$$

□

REFERENCES

- [1] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
- [2] G. E. Hutchinson, “The concept of pattern in ecology,” *Proceedings of the Academy of Natural Sciences of Philadelphia*, vol. 105, pp. 1–12, 1953.
- [3] H. Meinhardt, “Models of biological pattern formation,” *New York*, vol. 118, 1982.
- [4] U. Grenander, *Elements of pattern theory*. JHU Press, 1996.
- [5] M. Honarkhah and J. Caers, “Stochastic simulation of patterns using distance-based pattern modeling,” *Mathematical Geosciences*, vol. 42, no. 5, pp. 487–517, 2010.
- [6] D. Mumford and A. Desolneux, *Pattern theory: the stochastic analysis of real-world signals*. CRC Press, 2010.
- [7] P. Ball and N. R. Borley, *The self-made tapestry: pattern formation in nature*. Oxford University Press Oxford, 1999, vol. 198.
- [8] S. K. Reed, “Pattern recognition and categorization,” *Cognitive psychology*, vol. 3, no. 3, pp. 382–407, 1972.
- [9] N. Lund, *Attention and pattern recognition*. Psychology Press, 2001.
- [10] R. Pless and R. Souvenir, “A survey of manifold learning for images,” *IPSI Transactions on Computer Vision and Applications*, vol. 1, pp. 83–94, 2009.
- [11] J. Wang, *Geometric structure of high-dimensional data and dimensionality reduction*. Springer, 2012, vol. 5.
- [12] C. M. Drain, F. Nifatis, A. Vasenko, and J. D. Batteas, “Porphyrin tessellation by design: Metal-mediated self-assembly of large arrays and tapes,” *Angewandte Chemie International Edition*, vol. 37, no. 17, pp. 2344–2347, 1998.
- [13] H. Huang, P. Y. Mok, Y. Kwok, and J. Au, “Block pattern generation: From parameterizing human bodies to fit feature-aligned and flattenable 3d garments,” *Computers in Industry*, vol. 63, no. 7, pp. 680–691, 2012.
- [14] M. P. Winsor, “Starfish, jellyfish, and the order of life,” *Issues in*, 1976.
- [15] R. C. Moore and L. R. Laudon, *Evolution and classification of Paleozoic crinoids*. Geological Society of America, 1943, vol. 46.

- [16] B. B. Mandelbrot and B. B. Mandelbrot, *The fractal geometry of nature*. WH free-man New York, 1982, vol. 1.
- [17] W. James, *The principles of psychology*. Cosimo, Inc., 2007, vol. 1.
- [18] C. Chabris and D. J. Simons, *The invisible gorilla: And other ways our intuitions deceive us*. Harmony, 2010.
- [19] M. Halle, *The sound pattern of Russian*. De Gruyter Mouton, 2011.
- [20] J.-J. Nattiez, *Music and discourse: Toward a semiology of music*. Princeton University Press, 1990.
- [21] E. Weber, *Johann gottfried walther,'musicalisches lexicon oder musicalische bibliotheck'*, 2002.
- [22] J. Niessing and R. W. Friedrich, "Olfactory pattern classification by discrete neuronal network states," *Nature*, vol. 465, no. 7294, pp. 47–52, 2010.
- [23] K. A. Dahl and D. W. Oppo, "Sea surface temperature pattern reconstructions in the arabian sea," *Paleoceanography*, vol. 21, no. 1, 2006.
- [24] C. d. S. Pires, "Remark on the vectorlike nature of electromagnetism and electric charge quantization," *Physical Review D*, vol. 60, no. 7, p. 075 013, 1999.
- [25] W. O'Grady, M. Dobrovolsky, and F. Katamba, *Contemporary linguistics*. St. Martin's, 1997.
- [26] N. Chomsky, *Syntactic structures*. Walter de Gruyter, 2002.
- [27] P. N. Edwards, "Infrastructure and modernity: Force, time, and social organization in the history of sociotechnical systems," *Modernity and technology*, vol. 1, pp. 185–226, 2003.
- [28] H. V. Roberts, "Stock-market "patterns" and financial analysis: Methodological suggestions," *The Journal of Finance*, vol. 14, no. 1, pp. 1–10, 1959.
- [29] C. Darwin, *The origin of species*. PF Collier & son New York, 1909.
- [30] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.
- [31] D. Romer, *Advanced macroeconomics*. Mcgraw-hill, 2012.
- [32] R. S. Pindyck and D. L. Rubinfeld, *Microeconomics*. Pearson Education, 2014.

- [33] A. Einstein, “The general theory of relativity,” in *The Meaning of Relativity*, Springer, 1922, pp. 54–75.
- [34] A. Zeilinger, “Experiment and the foundations of quantum physics,” *More Things in Heaven and Earth*, pp. 482–498, 1999.
- [35] E. Meron, “Vegetation pattern formation: The mechanisms behind the forms,” *Physics Today*, vol. 72, no. 11, pp. 30–36, 2019.
- [36] F. Borgogno, P. D’Odorico, F. Laio, and L. Ridolfi, “Mathematical models of vegetation pattern formation in ecohydrology,” *Reviews of Geophysics*, vol. 47, no. 1, 2009.
- [37] B. T. Werner, “Complexity in natural landform patterns,” *Science*, vol. 284, no. 5411, pp. 102–104, 1999.
- [38] J. H. Brown, V. K. Gupta, B.-L. Li, B. T. Milne, C. Restrepo, and G. B. West, “The fractal nature of nature: Power laws, ecological complexity and biodiversity,” *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, vol. 357, no. 1421, pp. 619–626, 2002.
- [39] R. J. Baron, “Mechanisms of human facial recognition,” *International Journal of Man-Machine Studies*, vol. 15, no. 2, pp. 137–178, 1981.
- [40] W. Shugen, “Framework of pattern recognition model based on the cognitive psychology,” *Geo-spatial Information Science*, vol. 5, no. 2, pp. 74–78, 2002.
- [41] K. M. Galotti, *Cognitive psychology in and out of the laboratory*. Sage Publications, 2017.
- [42] P. V. Foukal, *Solar astrophysics*. John Wiley & Sons, 2008.
- [43] S. Wada, H. Kikura, and M. Aritomi, “Pattern recognition and signal processing of ultrasonic echo signal on two-phase flow,” *Flow Measurement and Instrumentation*, vol. 17, no. 4, pp. 207–224, 2006.
- [44] A. Wang *et al.*, “An industrial strength audio search algorithm,” in *Ismir*, Citeseer, vol. 2003, 2003, pp. 7–13.
- [45] C. Eivind and F. Drablos, “Detecting periodic patterns in biological sequences,” *Bioinformatics (Oxford, England)*, vol. 14, no. 6, pp. 498–507, 1998.
- [46] D. G. Kendall, D. Barden, T. K. Carne, and H. Le, *Shape and shape theory*. John Wiley & Sons, 2009, vol. 500.

- [47] D. Zhang and G. Lu, “Review of shape representation and description techniques,” *Pattern recognition*, vol. 37, no. 1, pp. 1–19, 2004.
- [48] T. Pavlidis, *Algorithms for graphics and image processing*. Springer Science & Business Media, 2012, p. 143.
- [49] P. Maragos, “Pattern spectrum and multiscale shape representation,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 11, no. 7, pp. 701–716, 1989.
- [50] M. Teague, “Image analysis via the general theory of moments,” *JOSA*, vol. 70, no. 8, pp. 920–930, 1980.
- [51] K. Yamada and K. Knight, “A syntax-based statistical translation model,” in *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, 2001, pp. 523–530.
- [52] E. H. Land, “Experiments in color vision,” *Scientific American*, vol. 200, no. 5, pp. 84–99, 1959.
- [53] ———, “The retinex theory of color vision,” *Scientific american*, vol. 237, no. 6, pp. 108–129, 1977.
- [54] J.-M. Morel, A. B. Petro, and C. Sbert, “Fast implementation of color constancy algorithms,” in *Color Imaging XIV: Displaying, Processing, Hardcopy, and Applications*, International Society for Optics and Photonics, vol. 7241, 2009, p. 724 106.
- [55] R. Kimmel, M. Elad, D. Shaked, R. Keshet, and I. Sobel, “A variational framework for retinex,” *International Journal of computer vision*, vol. 52, no. 1, pp. 7–23, 2003.
- [56] S. Zhang, T. Wang, J. Dong, and H. Yu, “Underwater image enhancement via extended multi-scale retinex,” *Neurocomputing*, vol. 245, pp. 1–9, 2017.
- [57] K. Schwarzschild, “On the gravitational field of a mass point according to einstein’s theory,” *arXiv preprint physics/9905030*, 1999.
- [58] J. Droste, “On the field of a single centre in einstein’s theory of gravitation,” *Koninklijke Nederlandse Akademie van Wetenschappen Proceedings Series B Physical Sciences*, vol. 17, pp. 998–1011, 1915.
- [59] D. Castelvechi, “Black hole pictured for first time-in spectacular detail,” *Nature*, vol. 568, no. 7752, pp. 284–285, 2019.

- [60] E. H. T. Collaboration *et al.*, “First m87 event horizon telescope results. i. the shadow of the supermassive black hole,” *arXiv preprint arXiv:1906.11238*, 2019.
- [61] I.-S. Liu, “On fourier’s law of heat conduction,” *Continuum mechanics and Thermodynamics*, vol. 2, no. 4, pp. 301–305, 1990.
- [62] J. Hadamard, “Sur les problèmes aux dérivées partielles et leur signification physique,” *Princeton university bulletin*, pp. 49–52, 1902.
- [63] J. Percus, “Equilibrium state of a classical fluid of hard rods in an external field,” *Journal of Statistical Physics*, vol. 15, no. 6, pp. 505–511, 1976.
- [64] J. Mawhin, *Critical point theory and Hamiltonian systems*. Springer Science & Business Media, 2013, vol. 74.
- [65] T. Kottos and U. Smilansky, “Periodic orbit theory and spectral statistics for quantum graphs,” *Annals of Physics*, vol. 274, no. 1, pp. 76–124, 1999.
- [66] Y. A. Kuznetsov, *Elements of applied bifurcation theory*. Springer Science & Business Media, 2013, vol. 112.
- [67] R. T. Glassey, “Finite-time blow-up for solutions of nonlinear wave equations,” *Mathematische Zeitschrift*, vol. 177, no. 3, pp. 323–340, 1981.
- [68] L. I. Rudin and S. Osher, “Total variation based image restoration with free local constraints,” in *Proceedings of 1st International Conference on Image Processing*, IEEE, vol. 1, 1994, pp. 31–35.
- [69] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.
- [70] D. B. Mumford and J. Shah, “Optimal approximations by piecewise smooth functions and associated variational problems,” *Communications on pure and applied mathematics*, 1989.
- [71] M. Bertalmio, V. Caselles, E. Provenzi, and A. Rizzi, “Perceptual color correction through variational techniques,” *IEEE Transactions on Image Processing*, vol. 16, no. 4, pp. 1058–1072, 2007.
- [72] H. Weyl, *Symmetry*. Princeton University Press, 2015, vol. 104.
- [73] B. C. Van Fraassen, “Laws and symmetry,” 1989.

- [74] Y. Liu, R. T. Collins, and Y. Tsin, “A computational model for periodic pattern perception based on frieze and wallpaper groups,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 3, pp. 354–371, 2004.
- [75] E. Hitzer and D. Ichikawa, “Representation of crystallographic subperiodic groups in clifford’s geometric algebra,” *Advances in Applied Clifford Algebras*, vol. 23, no. 4, pp. 887–906, 2013.
- [76] H. Hiller, “Crystallography and cohomology of groups,” *The American Mathematical Monthly*, vol. 93, no. 10, pp. 765–779, 1986.
- [77] C. J. Mulvey, “A generalisation of gelfand duality,” *Journal of Algebra*, vol. 56, no. 2, pp. 499–505, 1979.
- [78] J. M. Lee, “Smooth manifolds,” in *Introduction to Smooth Manifolds*, Springer, 2013, pp. 1–31.
- [79] J. E. Whitesitt, *Boolean algebra and its applications*. Courier Corporation, 2012.
- [80] S. Awodey, *Category theory*. Oxford university press, 2010.
- [81] M. Barr and C. Wells, *Category theory for computing science*. Prentice Hall New York, 1990, vol. 49.
- [82] M. D. Resnik, *Mathematics as a Science of Patterns*. Oxford University Press, 1997.
- [83] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, 2. MIT press Cambridge, 2016, vol. 1.
- [84] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [85] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [86] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.
- [87] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

- [88] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [89] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *arXiv preprint arXiv:1712.09913*, 2017.
- [90] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International conference on machine learning*, PMLR, 2013, pp. 1310–1318.
- [91] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, “Dying relu and initialization: Theory and numerical examples,” *arXiv preprint arXiv:1903.06733*, 2019.
- [92] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, “Co-teaching: Robust training of deep neural networks with extremely noisy labels,” *arXiv preprint arXiv:1804.06872*, 2018.
- [93] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [94] T. Salimans and D. P. Kingma, “Weight normalization: A simple reparameterization to accelerate training of deep neural networks,” *arXiv preprint arXiv:1602.07868*, 2016.
- [95] L. Bieberbach, “Über die Bewegungsgruppen der Euklidischen Räume,” *Mathematische Annalen*, vol. 70, no. 3, pp. 297–336, 1911.
- [96] E. Fedorov, “Symmetry in the plane,” in *Zapiski Imperatorskogo S. Peterburgskogo Mineralogicheskogo Obshchestva [Proc. S. Peterb. Mineral. Soc.]*, vol. 2, 1891, pp. 345–390.
- [97] H. Hiller, “Crystallography and cohomology of groups,” *The American Mathematical Monthly*, vol. 93, no. 10, pp. 765–779, 1986.
- [98] Y. Liu, R. Collins, and Y. Tsin, “A computational model for periodic pattern perception based on frieze and wallpaper groups,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 3, pp. 354–371, 2004.
- [99] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.
- [100] M. Jana and C. Rao, “Two-dimensional inorganic analogues of graphene: Transition metal dichalcogenides,” *Phil. Trans. R. Soc. A*, vol. 374, no. 2076, p. 20150318, 2016.

- [101] D. Voiry, A. Mohite, and M. Chhowalla, “Phase engineering of transition metal dichalcogenides,” *Chemical Society Reviews*, vol. 44, no. 9, pp. 2702–2712, 2015.
- [102] B. Berkels, A. Rätz, M. Rumpf, and A. Voigt, “Identification of grain boundary contours at atomic scale,” in *International Conference on Scale Space and Variational Methods in Computer Vision*, Springer, 2007, pp. 765–776.
- [103] ———, “Extracting grain boundaries and macroscopic deformations from images on atomic scale,” *Journal of Scientific Computing*, vol. 35, no. 1, pp. 1–23, 2008.
- [104] M. Boerdgen, B. Berkels, M. Rumpf, and D. Cremers, “Convex relaxation for grain segmentation at atomic scale,” in *Vision, Modeling, and Visualization*, 2010, pp. 179–186.
- [105] P. Hirvonen, G. M. La Boissonière, Z. Fan, C. V. Achim, N. Provatas, K. R. Elder, and T. Ala-Nissila, “Grain extraction and microstructural analysis method for two-dimensional poly and quasicrystalline solids,” *Physical Review Materials*, vol. 2, no. 10, p. 103603, 2018.
- [106] J. Lu and H. Yang, “Phase-space sketching for crystal image analysis based on synchrosqueezed transforms,” *SIAM Journal on Imaging Sciences*, vol. 11, no. 3, pp. 1954–1978, 2018.
- [107] D. Zosso, K. Dragomiretskiy, A. L. Bertozzi, and P. S. Weiss, “Two-dimensional compact variational mode decomposition,” *Journal of Mathematical Imaging and Vision*, vol. 58, no. 2, pp. 294–320, 2017.
- [108] W. Friedrich, P. Knipping, and M. Laue, “Interferenzerscheinungen bei Röntgenstrahlen,” *Annalen der Physik*, vol. 346, no. 10, pp. 971–988, 1913.
- [109] G. Thomson and A. Reid, “Diffraction of cathode rays by a thin film,” *Nature*, vol. 119, no. 3007, p. 890, 1927.
- [110] D. Petersen and D. Middleton, “Sampling and reconstruction of wave-number-limited functions in N-dimensional Euclidean spaces,” *Information and Control*, vol. 5, no. 4, pp. 279–323, 1962.
- [111] H. Wu, K. Malafant, L. Pendridge, P. Sharpe, and J. Walker, “Simulation of two-dimensional point patterns: Application of a lattice framework approach,” *Ecological Modelling*, vol. 38, no. 3-4, pp. 299–308, 1987.
- [112] W. Haynes, *CRC Handbook of Chemistry and Physics*. CRC Press, 2014.
- [113] W. Davey, “Precision measurements of the lattice constants of twelve common metals,” *Physical Review*, vol. 25, no. 6, p. 753, 1925.

- [114] T. Matsuyama, S. Miura, and M. Nagao, “Structural analysis of natural textures by Fourier transformation,” *Computer Vision, Graphics, and Image Processing*, vol. 24, no. 3, pp. 347–362, 1983.
- [115] M. Park, K. Brocklehurst, R. Collins, and Y. Liu, “Deformed lattice detection in real-world images using mean-shift belief propagation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1804–1816, 2009.
- [116] F. Schaffalitzky and A. Zisserman, “Geometric grouping of repeated elements within images,” in *Shape, Contour and Grouping in Computer Vision*, Springer, 1999, pp. 165–181.
- [117] J. Hays, M. Leordeanu, A. Efros, and Y. Liu, “Discovering texture regularity as a higher-order correspondence problem,” in *European Conference on Computer Vision*, Springer, 2006, pp. 522–535.
- [118] Y. He and S. H. Kang, “Lattice identification and separation: Theory and algorithm,” *arXiv:1901.02520*, 2018.
- [119] J. Hock and R. McQuistan, “The occupation statistics for indistinguishable dumbbells on a $2 \times 2 \times N$ lattice space,” *Journal of Mathematical Physics*, vol. 24, no. 7, pp. 1859–1865, 1983.
- [120] C. Rogers, “Mean values over the space of lattices,” *Acta Mathematica*, vol. 94, no. 1, pp. 249–287, 1955.
- [121] L. Schiff, “Lattice-space quantization of a nonlinear field theory,” *Physical Review*, vol. 92, no. 3, p. 766, 1953.
- [122] B. Vallée and A. Vera, “Probabilistic analyses of lattice reduction algorithms,” in *The LLL Algorithm*, Springer, 2009, pp. 71–143.
- [123] H. Farkas and I. Kra, “Riemann surfaces,” in *Riemann Surfaces*, Springer, 1992, pp. 9–31.
- [124] C. Siegel, *Lectures on the Geometry of Numbers*. Springer Science & Business Media, 2013.
- [125] G. Airy, “On the diffraction of an object-glass with circular aperture,” *Transactions of the Cambridge Philosophical Society*, vol. 5, p. 283, 1835.
- [126] R. Bracewell, “Strip integration in radio astronomy,” *Australian Journal of Physics*, vol. 9, no. 2, pp. 198–217, 1956.

- [127] D. Burago, Y. Burago, and S. Ivanov, *A Course in Metric Geometry*. American Mathematical Soc., 2001, vol. 33.
- [128] H. Minkowski, “Geometrie der zahlen,” *Bulletin of American Mathematical Society*, vol. 21, no. 3, pp. 131–132, 1914.
- [129] —, “Üeber die positiven quadratischen Formen und über kettenbruchähnliche Algorithmen,” *Journal für die Reine und Angewandte Mathematik*, vol. 107, pp. 278–297, 1891.
- [130] C. Hermite and E. Picard, *Oeuvres de Charles Hermite*. Gauthier-Villars, 1908, vol. 2.
- [131] J. Cassels, *Rational Quadratic Forms*. Courier Dover Publications, 2008.
- [132] A. Korkine and G. Zolotareff, “Sur les formes quadratiques,” *Mathematische Annalen*, vol. 6, no. 3, pp. 366–389, 1873.
- [133] A. Lenstra, H. Lenstra, and L. Lovász, “Factoring polynomials with rational coefficients,” *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, 1982.
- [134] M. Ajtai, “The shortest vector problem in L_2 is NP-hard for randomized reductions,” in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, ACM, 1998, pp. 10–19.
- [135] T. M. Apostol, *Modular Functions and Dirichlet Series in Number Theory*. Springer Science & Business Media, 2012, vol. 41.
- [136] R. Alperin, “Notes: $\mathrm{PSL}_2(\mathbb{Z}) = \mathbb{Z}_2 * \mathbb{Z}_3$,” *Amer.Math.Monthly*, vol. 100, no. 4, pp. 385–386, 1993.
- [137] C. Johansson and J. Linde, “Röntgenographische Bestimmung der Atomanordnung in den Mischkristallreihen Au-Cu und Pd-Cu,” *Annalen der Physik*, vol. 383, no. 21, pp. 439–460, 1925.
- [138] T. Harman, P. Taylor, M. Walsh, and B. LaForge, “Quantum dot superlattice thermoelectric materials and devices,” *Science*, vol. 297, no. 5590, pp. 2229–2232, 2002.
- [139] T. Musho and D. Walker, “Thermoelectric properties of superlattice materials with variably spaced layers,” *Journal of Materials Research*, vol. 26, no. 15, pp. 1993–2000, 2011.

- [140] P. Yashar, S. Barnett, J. Rechner, and W. Sproul, "Structure and mechanical properties of polycrystalline CrN/TiN superlattices," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 16, no. 5, pp. 2913–2918, 1998.
- [141] A. Kudrolli, B. Pier, and J. Gollub, "Superlattice patterns in surface waves," *Physica D: Nonlinear Phenomena*, vol. 123, no. 1-4, pp. 99–111, 1998.
- [142] M. Silber, C. Topaz, and A. Skeldon, "Two-frequency forced faraday waves: Weakly damped modes and pattern selection," *Physica D: Nonlinear Phenomena*, vol. 143, no. 1-4, pp. 205–225, 2000.
- [143] E. Westhoff, V. Kneisel, Y. Logvin, T. Ackemann, and W. Lange, "Pattern formation in the presence of an intrinsic polarization instability," *Journal of Optics B: Quantum and Semiclassical Optics*, vol. 2, no. 3, p. 386, 2000.
- [144] W. Choi, N. Choudhary, G. Han, J. Park, D. Akinwande, and Y. Lee, "Recent development of two-dimensional transition metal dichalcogenides and their applications," *Materials Today*, vol. 20, no. 3, pp. 116–130, 2017.
- [145] K. Novoselov, A. Geim, S. Morozov, D. Jiang, Y. Zhang, S. Dubonos, I. Grigorieva, and A. Firsov, "Electric field effect in atomically thin carbon films," *Science*, vol. 306, no. 5696, pp. 666–669, 2004.
- [146] C. Rao, U. Maitra, and U. Waghmare, "Extraordinary attributes of 2-dimensional MoS_2 nanosheets," *Chemical Physics Letters*, vol. 609, pp. 172–183, 2014.
- [147] A. Eftekhari, "Tungsten dichalcogenides (WS_2 , WSe_2 , and WTe_2): Materials chemistry and applications," *Journal of Materials Chemistry A*, vol. 5, no. 35, pp. 18 299–18 325, 2017.
- [148] B. Julesz, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, no. 5802, p. 91, 1981.
- [149] T. Leung and J. Malik, "Detecting, localizing and grouping repeated scene elements from an image," in *European Conference on Computer Vision*, Springer, 1996, pp. 546–555.
- [150] I. Amidror, *The Theory of the Moiré Phenomenon: Volume I: Periodic Layers*. Springer Science & Business Media, 2009, vol. 38.
- [151] G. Bassett, J. Menter, and D. Pashley, "Moiré patterns on electron micrographs, and their application to the study of dislocations in metals," *Proc. R. Soc. Lond. A*, vol. 246, no. 1246, pp. 345–368, 1958.

- [152] M. Gustafsson, "Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy," *Journal of Microscopy*, vol. 198, no. 2, pp. 82–87, 2000.
- [153] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, IEEE, vol. 1, 2001, pp. 105–112.
- [154] C. G. Healey and J. T. Enns, "Building perceptual textures to visualize multidimensional datasets," in *Proceedings Visualization'98 (Cat. No. 98CB36276)*, IEEE, 1998, pp. 111–118.
- [155] C. Weigle, W. Emigh, G. Liu, R. Taylor, J. Enns, and C. Healey, "Oriented texture slivers: A technique for local value estimation of multiple scalar fields," in *Proceedings Graphics Interface*, 2000, pp. 163–170.
- [156] J. Wolfe, K. Cave, and S. Franzel, "Guided search: An alternative to the feature integration model for visual search.," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 15, no. 3, p. 419, 1989.
- [157] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [158] D. Naidu and R. Fisher, "A comparative analysis of algorithms for determining the peak position of a stripe to sub-pixel accuracy," in *BMVC91*, Springer, 1991, pp. 217–225.
- [159] C. Sun, "Fast optical flow using 3D shortest path techniques," *Image and Vision Computing*, vol. 20, no. 13-14, pp. 981–991, 2002.
- [160] H. Nobach and M. Honkanen, "Two-dimensional Gaussian regression for sub-pixel displacement estimation in particle image velocimetry or particle position estimation in particle tracking velocimetry," *Experiments in Fluids*, vol. 38, no. 4, pp. 511–515, 2005.
- [161] S. Horbelt, M. Liebling, and M. Unser, "Discretization of the Radon transform and of its inverse by spline convolutions," *IEEE Transactions on Medical Imaging*, vol. 21, no. 4, pp. 363–376, 2002.
- [162] A. Bonissent and F. Abraham, "Application of perturbation theory to the crystal–melt interface," *The Journal of Chemical Physics*, vol. 74, no. 2, pp. 1306–1309, 1981.

- [163] J. Zheng, H. Zhang, S. Dong, Y. Liu, C. Nai, H. Shin, H. Jeong, B. Liu, and K. Loh, “High yield exfoliation of two-dimensional chalcogenides using sodium naphthalenide,” *Nature Communications*, vol. 5, p. 2995, 2014.
- [164] C. Rao, M. Ramakrishna, and U. Maitra, “Graphene analogues of inorganic layered materials,” *Angewandte Chemie International Edition*, vol. 52, no. 50, pp. 13 162–13 185, 2013.
- [165] Y. He and S. H. Kang, “Lattice metric space application to grain defect detection,” in *International Conference on Scale Space and Variational Methods in Computer Vision*, Springer, 2019.
- [166] L. Biró and P. Lambin, “Grain boundaries in graphene grown by chemical vapor deposition,” *New Journal of Physics*, vol. 15, no. 3, p. 035 024, 2013.
- [167] C. Kittel, P. McEuen, and P. McEuen, *Introduction to solid state physics*. Wiley New York, 1996, vol. 8.
- [168] N. Mevenkamp and B. Berkels, “Variational multi-phase segmentation using high-dimensional local features,” in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2016, pp. 1–9.
- [169] H. Singer and I. Singer, “Analysis and visualization of multiply oriented lattice structures by a two-dimensional continuous wavelet transform,” *Physical Review E*, vol. 74, no. 3, p. 031 103, 2006.
- [170] I. Daubechies, “A nonlinear squeezing of the continuous wavelet transform based on auditory nerve models,” *Wavelets in medicine and biology*, pp. 527–546, 1996.
- [171] H. Yang, J. Lu, and L. Ying, “Crystal image analysis using 2D synchrosqueezed transforms,” *Multiscale Model Simul.*, vol. 13, no. 4, pp. 1542–1572, 2015.
- [172] E. A. Lazar, J. Han, and D. J. Srolovitz, “Topological framework for local structure analysis in condensed matter,” *Proceedings of the National Academy of Sciences*, vol. 112, no. 43, E5769–E5776, 2015.
- [173] A. Stukowski, “Structure identification methods for atomistic simulations of crystalline materials,” *Model. Simul. Mater. Sci. Eng.*, vol. 20, no. 4, p. 045 021, 2012.
- [174] G. M. La Boissoniere and R. Choksi, “Atom based grain extraction and measurement of geometric properties,” *Model Simul. Mater. Sci. Eng.*, vol. 26, no. 3, p. 035 001, 2018.

- [175] D. Arthur and S. Vassilvitskii, “K-means++: The advantages of careful seeding,” in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.
- [176] S. H. Kang, B. Sandberg, and A. M. Yip, “A regularized k-means and multiphase scale segmentation,” *Inverse Problems & Imaging*, vol. 5, no. 2, pp. 407–429, 2011.
- [177] L. Rokach and O. Maimon, “Clustering methods,” in *Data mining and knowledge discovery handbook*, Springer, 2005, pp. 321–352.
- [178] P. Y. Huang, C. S. Ruiz-Vargas, A. M. van der Zande, W. S. Whitney, M. P. Levendoff, J. W. Kevek, S. Garg, J. S. Alden, C. J. Hustedt, Y. Zhu, *et al.*, “Grains and grain boundaries in single-layer graphene atomic patchwork quilts,” *Nature*, vol. 469, no. 7330, p. 389, 2011.
- [179] D. Medlin, K. Hattar, J. Zimmerman, F. Abdeljawad, and S. Foiles, “Defect character at grain boundary facet junctions: Analysis of an asymmetric $\Sigma=5$ grain boundary in Fe,” *Acta Materialia*, vol. 124, pp. 383–396, 2017.
- [180] T. Radetic, F. Lancon, and U. Dahmen, “Chevron defect at the intersection of grain boundaries with free surfaces in Au,” *Physical review letters*, vol. 89, no. 8, p. 085 502, 2002.
- [181] J. Toriwaki, T. Saitoh, and M. Okada, “Distance transformation and skeleton for shape feature analysis,” in *Visual Form*, Springer, 1992, pp. 547–563.
- [182] T. Sebastian and B. Kimia, “Curves vs. skeletons in object recognition,” *Signal processing*, vol. 85, no. 2, pp. 247–263, 2005.
- [183] X. Yang, X. Bai, D. Yu, and L. Latecki, “Shape classification based on skeleton path similarity,” in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Springer, 2007, pp. 375–386.
- [184] N. Chapman and J. Chapman, *Digital multimedia*. Wiley Publishing, 2009, p. 86.
- [185] U. Montanari, “Continuous skeletons from digitized images,” *Journal of the ACM (JACM)*, vol. 16, no. 4, pp. 534–549, 1969.
- [186] H. Blum, “A transformation for extracting new descriptors of shape,” *Models for the perception of speech and visual form*, vol. 19, no. 5, pp. 362–380, 1967.
- [187] H. Blum and R. Nagel, “Shape description using weighted symmetric axis features,” *Pattern recognition*, vol. 10, no. 3, pp. 167–180, 1978.

- [188] U. Montanari, "A method for obtaining skeletons using a quasi-Euclidean distance," *Journal of the ACM (JACM)*, vol. 15, no. 4, pp. 600–624, 1968.
- [189] A. Frank, J. Daniels, and D. Unangst, "Progressive image transmission using a growth-geometry coding," *Proceedings of the IEEE*, vol. 68, no. 7, pp. 897–909, 1980.
- [190] L. Calabi and W. Hartnett, "Shape recognition, prairie fires, convex deficiencies and skeletons," *The American Mathematical Monthly*, vol. 75, no. 4, pp. 335–342, 1968.
- [191] k. Siddiqi, s. Bouix, A. Tannenbaum, and S. Zucker, "Hamilton-Jacobi skeletons," *International Journal of Computer Vision*, vol. 48, no. 3, pp. 215–231, 2002.
- [192] J. P. Balarini and S. Nesmachnow, "A C++ Implementation of Otsu's Image Segmentation Method," *Image Processing On Line*, vol. 6, pp. 155–164, 2016.
- [193] H. Choi, S. Choi, and H. Moon, "Mathematical theory of medial axis transform," *pacific journal of mathematics*, vol. 181, no. 1, pp. 57–88, 1997.
- [194] J. W. Brandt and V. R. Algazi, "Continuous skeleton computation by voronoi diagram," *CVGIP: Image understanding*, vol. 55, no. 3, pp. 329–338, 1992.
- [195] R. L. Ogniewicz and M. Ilg, "Voronoi skeletons: Theory and applications.," in *CVPR*, vol. 92, 1992, pp. 63–69.
- [196] F. Leymarie and M. D. Levine, "Simulating the grassfire transform using an active contour model," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 1, pp. 56–75, 1992.
- [197] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, "Shapes, shocks, and deformations i: The components of two-dimensional shape and the reaction-diffusion space," *International journal of computer vision*, vol. 15, no. 3, pp. 189–224, 1995.
- [198] C. Arcelli and G. S. Di Baja, "A width-independent fast thinning algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 4, pp. 463–474, 1985.
- [199] I. Bitter, A. E. Kaufman, and M. Sato, "Penalized-distance volumetric skeleton algorithm," *IEEE Transactions on Visualization and computer Graphics*, vol. 7, no. 3, pp. 195–206, 2001.
- [200] P. K. Saha, G. Borgefors, and G. S. di Baja, "A survey on skeletonization algorithms and their applications," *Pattern recognition letters*, vol. 76, pp. 3–12, 2016.

- [201] S. Nathan and P. Kansal, "Skeletonnet: Shape pixel to skeleton pixel," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.
- [202] H. Zhao, "A fast sweeping method for Eikonal equations," *Mathematics of computation*, vol. 74, no. 250, pp. 603–627, 2005.
- [203] H. Zhao, S. Osher, B. Merriman, and M. Kang, "Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method," *Computer Vision and Image Understanding*, vol. 80, no. 3, pp. 295–314, 2000.
- [204] M. G. Crandall and P.-L. Lions, "Viscosity solutions of hamilton-jacobi equations," *Transactions of the American mathematical society*, vol. 277, no. 1, pp. 1–42, 1983.
- [205] E. Rouy and A. Tourin, "A viscosity solutions approach to shape-from-shading," *SIAM Journal on Numerical Analysis*, vol. 29, no. 3, pp. 867–884, 1992.
- [206] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, "2d euclidean distance transform algorithms: A comparative survey," *ACM Computing Surveys (CSUR)*, vol. 40, no. 1, pp. 1–44, 2008.
- [207] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory of computing*, vol. 8, no. 1, pp. 415–428, 2012.
- [208] J. August, A. Tannenbaum, and S. Zucker, "On the evolution of the skeleton," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, vol. 1, 1999, pp. 315–322.
- [209] X. Bai, L. Latecki, and W. Liu, "Skeleton pruning by contour partitioning with discrete curve evolution," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 3, pp. 449–462, 2007.
- [210] I. Kunttu, L. Lepisto, J. Rauhamaa, and A. Visa, "Multiscale Fourier descriptor for shape classification," in *12th International Conference on Image Analysis and Processing, 2003. Proceedings.*, IEEE, 2003, pp. 536–541.
- [211] M. Levandowsky and D. Winter, "Distance between sets," *Nature*, vol. 234, no. 5323, pp. 34–35, 1971.
- [212] T. Sørensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons," 1948.
- [213] M. Cardoso, T. Arbel, S. Lee, V. Cheplygina, S. Balocco, D. Mateus, G. Zahnd, L. Maier-Hein, S. Demirci, E. Granger, and L. Duong, "Intravascular imaging

and computer assisted stenting, and large-scale annotation of biomedical data and expert label synthesis,” in *CVII-STENT and Second International Workshop, LABELS*, Springer, 2017.

- [214] G. Matheron, *Random sets and integral geometry*. Wiley New York, 1974, xxiii, 261 p. ISBN: 0471576212.
- [215] F. Attneave, “Some informational aspects of visual perception,” *Psychological review*, vol. 61, no. 3, p. 183, 1954.
- [216] P. Monasse and F. Guichard, “Scale-space from a level lines tree,” *Journal of Visual Communication and Image Representation*, vol. 11, no. 2, pp. 224–236, 2000.
- [217] L. Ambrosio, V. Caselles, S. Masnou, and J.-M. Morel, “Connected components of sets of finite perimeter and applications to image processing,” *Journal of the European Mathematical Society*, vol. 3, no. 1, pp. 39–92, 2001.
- [218] F. Cao, J.-L. Lisani, J.-M. Morel, P. Musé, and F. Sur, *A theory of shape identification*. Springer Science & Business Media, 2008.
- [219] U. Montanari, “A note on minimal length polygonal approximation to a digitized contour,” *Communications of the ACM*, vol. 13, no. 1, pp. 41–47, 1970.
- [220] M. E. Mortenson, *Mathematics for computer graphics applications*. Industrial Press Inc., 1999.
- [221] C. Nadal, R. Legault, and C. Y. Suen, “Complementary algorithms for the recognition of totally unconstrained handwritten numerals,” in *[1990] Proceedings. 10th International Conference on Pattern Recognition*, IEEE, vol. 1, 1990, pp. 443–449.
- [222] A. Kirsanov, A. Vavilin, and K. Jo, “Contour-based algorithm for vectorization of satellite images,” in *International Forum on Strategic Technology 2010*, IEEE, 2010, pp. 241–245.
- [223] H.-M. Yang, J.-J. Lu, and H.-J. Lee, “A bezier curve-based approach to shape description for chinese calligraphy characters,” in *Proceedings of Sixth International Conference on Document Analysis and Recognition*, IEEE, 2001, pp. 276–280.
- [224] K. Tombre and S. Tabbone, “Vectorization in graphics recognition: To thin or not to thin,” in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, IEEE, vol. 2, 2000, pp. 91–96.
- [225] J. J. Zou and H. Yan, “Cartoon image vectorization based on shape subdivision,” in *Proceedings. Computer Graphics International 2001*, IEEE, 2001, pp. 225–231.

- [226] H.-H. Chang and H. Yan, "Vectorization of hand-drawn image using piecewise cubic bezier curves fitting," *Pattern recognition*, vol. 31, no. 11, pp. 1747–1755, 1998.
- [227] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer graphics and image processing*, vol. 1, no. 3, pp. 244–256, 1972.
- [228] L. Cinque, S. Levialdi, and A. Malizia, "Shape description using cubic polynomial bezier curves," *Pattern Recognition Letters*, vol. 19, no. 9, pp. 821–828, 1998.
- [229] A. S. Montero and J. Lang, "Skeleton pruning by contour approximation and the integer medial axis transform," *Computers & Graphics*, vol. 36, no. 5, pp. 477–487, 2012.
- [230] S. Pal, P. Ganguly, and P. Biswas, "Cubic b  zier approximation of a digitized curve," *Pattern recognition*, vol. 40, no. 10, pp. 2730–2741, 2007.
- [231] W. Pan, Z. Lian, Y. Tang, and J. Xiao, "Skeleton-guided vectorization of chinese calligraphy images," in *2014 IEEE 16th International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2014, pp. 1–6.
- [232] M. Sarfraz, "Vectorizing outlines of generic shapes by cubic spline using simulated annealing," *International Journal of Computer Mathematics*, vol. 87, no. 8, pp. 1736–1751, 2010.
- [233] D. Chetverikov, "A simple and efficient algorithm for detection of high curvature points in planar curves," in *International Conference on Computer Analysis of Images and Patterns*, Springer, 2003, pp. 746–753.
- [234] L. Alvarez and J. M. Morel, "Formalization and computational aspects of image analysis," *Acta numerica*, vol. 3, pp. 1–59, 1994.
- [235] G. Sapiro and A. Tannenbaum, "Affine invariant scale-space," *International journal of computer vision*, vol. 11, no. 1, pp. 25–44, 1993.
- [236] L. Alvarez and F. Morales, "Affine morphological multiscale analysis of corners and multiple junctions," *International Journal of Computer Vision*, vol. 25, no. 2, pp. 95–107, 1997.
- [237] L. Alvarez, "Corner detection using the affine morphological scale space," in *International Conference on Scale Space and Variational Methods in Computer Vision*, Springer, 2017, pp. 29–40.
- [238] L.   lvarez, F. Guichard, P.-L. Lions, and J.-M. Morel, "Axiomes et   quations fondamentales du traitement d'images.(analyse multi  chelle et edp)," *Comptes rendus*

- de l'Académie des sciences. Série 1, Mathématique*, vol. 315, no. 2, pp. 135–138, 1992.
- [239] L. Moisan, “Affine plane curve evolution: A fully consistent scheme,” *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 411–420, 1998.
 - [240] A. P. Witkin, “Scale-space filtering,” in *Readings in Computer Vision*, Elsevier, 1987, pp. 329–332.
 - [241] J. Weickert, S. Ishikawa, and A. Imiya, “Linear scale-space has first been proposed in japan,” *Journal of Mathematical Imaging and Vision*, vol. 10, no. 3, pp. 237–252, 1999.
 - [242] T. Iijima, “Basis theory on the normalization of two-dimensional visual pattern, studies on information and control, pattern recognition issue,” *IEICE Japan*, vol. 1, 1963.
 - [243] Y. He, S. H. Kang, and J.-M. Morel, “Silhouette vectorization by affine scale-space,” *arXiv preprint arXiv:2007.12117*, 2020.
 - [244] A. Ciomaga, P. Monasse, and J. M. Morel, “Level lines shortening yields an image curvature microscope,” in *2010 IEEE International Conference on Image Processing*, IEEE, 2010, pp. 4129–4132.
 - [245] F. Cao, *Geometric curve evolution and image processing*. Springer Science & Business Media, 2003.
 - [246] V. Caselles and P. Monasse, *Geometric description of images as topographic maps*. Springer, 2009.
 - [247] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.
 - [248] J.-M. Morel and G. Yu, “Asift: A new framework for fully affine invariant image comparison,” *SIAM journal on imaging sciences*, vol. 2, no. 2, pp. 438–469, 2009.
 - [249] A. Ciomaga, P. Monasse, and J.-M. Morel, “The image curvature microscope: Accurate curvature computation at subpixel resolution,” *Image Processing On Line*, vol. 7, pp. 197–217, 2017.
 - [250] A. Andrew, “Another efficient algorithm for convex hulls in two dimensions,” *Information Processing Letters*, vol. 9, no. 5, pp. 216–219, 1979.

- [251] F. P. Preparata and M. I. Shamos, *Computational geometry: an introduction*. Springer Science & Business Media, 2012.
- [252] M. Plass and M. Stone, “Curve-fitting with piecewise parametric cubics,” in *Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, 1983, pp. 229–239.
- [253] *SVG SILH*, <https://svgsilh.com>, All contents are released under Creative Commons CC0.
- [254] M. Goldapp, “Approximation of circular arcs by cubic polynomials,” *Computer Aided Geometric Design*, vol. 8, no. 3, pp. 227–238, 1991.
- [255] C. G. Harris, M. Stephens, *et al.*, “A combined corner and edge detector,” in *Alvey vision conference*, Citeseer, vol. 15, 1988, pp. 10–5244.
- [256] E. Rosten and T. Drummond, “Fusing points and lines for high performance tracking,” in *Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1*, Ieee, vol. 2, 2005, pp. 1508–1515.
- [257] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *European conference on computer vision*, Springer, 2006, pp. 404–417.
- [258] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.
- [259] C. Schmid, R. Mohr, and C. Bauckhage, “Evaluation of interest point detectors,” *International Journal of computer vision*, vol. 37, no. 2, pp. 151–172, 2000.
- [260] *Vector Magic*, <https://vectormagic.com>.
- [261] *Inkspace*, <https://inkscape.org>.
- [262] *Adobe Illustrator*, <https://www.adobe.com/products/illustrator.html>.
- [263] D. Khan, M. A. Shirazi, and M. Y. Kim, “Single shot laser speckle based 3D acquisition system for medical applications,” *Optics and Lasers in Engineering*, vol. 105, pp. 43–53, 2018.
- [264] G. Casciola, D. Lazzaro, L. B. Montefusco, and S. Morigi, “Shape preserving surface reconstruction using locally anisotropic radial basis function interpolants,” *Computers & Mathematics with Applications*, vol. 51, no. 8, pp. 1185–1198, 2006.

- [265] F. Calakli and G. Taubin, “SSD: Smooth signed distance surface reconstruction,” in *Computer Graphics Forum*, Wiley Online Library, vol. 30, 2011, pp. 1993–2002.
- [266] Z. Bi and L. Wang, “Advances in 3D data acquisition and processing for industrial applications,” *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 5, pp. 403–413, 2010.
- [267] L. Gomes, O. R. P. Bellon, and L. Silva, “3D reconstruction methods for digital preservation of cultural heritage: A survey,” *Pattern Recognition Letters*, vol. 50, pp. 3–14, 2014.
- [268] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, “Point set surfaces,” in *Proceedings of the Conference on Visualization’01*, IEEE Computer Society, 2001, pp. 21–28.
- [269] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations,” *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988.
- [270] Y. Shi and W. C. Karl, “Shape reconstruction from unorganized points with a data-driven level set method,” in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, IEEE, vol. 3, 2004, pp. iii–13.
- [271] J. Haličková and K. Mikula, “Level set method for surface reconstruction and its application in surveying,” *Journal of Surveying Engineering*, vol. 142, no. 3, p. 04 016 007, 2016.
- [272] H. Liu, Z. Yao, S. Leung, and T. F. Chan, “A level set based variational principal flow method for nonparametric dimension reduction on Riemannian manifolds,” *SIAM Journal on Scientific Computing*, vol. 39, no. 4, A1616–A1646, 2017.
- [273] J. Liang, F. Park, and H.-K. Zhao, “Robust and efficient implicit surface reconstruction for point clouds based on convexified image segmentation,” *Journal of Scientific Computing*, vol. 54, no. 2-3, pp. 577–602, 2013.
- [274] V. Estellers, D. Zosso, R. Lai, S. Osher, J.-P. Thiran, and X. Bresson, “Efficient algorithm for level set method preserving distance function,” *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4722–4734, 2012.
- [275] M. Wan, Y. Wang, E. Bae, X.-C. Tai, and D. Wang, “Reconstructing open surfaces via graph-cuts,” *IEEE transactions on visualization and computer graphics*, vol. 19, no. 2, pp. 306–318, 2012.

- [276] H. Liu, X. Wang, and W. Qiang, “Implicit surface reconstruction from 3D scattered points based on variational level set method,” in *2008 2nd International Symposium on Systems and Control in Aerospace and Astronautics*, IEEE, 2008, pp. 1–5.
- [277] R. Lai, X.-C. Tai, and T. F. Chan, “A ridge and corner preserving model for surface restoration,” *SIAM Journal on Scientific Computing*, vol. 35, no. 2, A675–A695, 2013.
- [278] H. Li, Y. Li, R. Yu, J. Sun, and J. Kim, “Surface reconstruction from unorganized points with ℓ_0 gradient minimization,” *Computer Vision and Image Understanding*, vol. 169, pp. 108–118, 2018.
- [279] H.-K. Zhao, S. Osher, and R. Fedkiw, “Fast surface reconstruction using the level set method,” in *Proceedings IEEE Workshop on Variational and Level Set Methods in Computer Vision*, IEEE, 2001, pp. 194–201.
- [280] P. Smereka, “Semi-implicit level set methods for curvature and surface diffusion motion,” *Journal of Scientific Computing*, vol. 19, no. 1, pp. 439–456, 2003.
- [281] X. Bresson, S. Esedoglu, P. Vanderghelynst, J.-P. Thiran, and S. Osher, “Fast global minimization of the active contour/snake model,” *Journal of Mathematical Imaging and Vision*, vol. 28, no. 2, pp. 151–167, 2007.
- [282] J. Shi, M. Wan, X.-C. Tai, and D. Wang, “Curvature minimization for surface reconstruction with features,” in *International Conference on Scale Space and Variational Methods in Computer Vision*, Springer, 2011, pp. 495–507.
- [283] Y. He, M. Huska, S. H. Kang, and H. Liu, “Fast algorithms for surface reconstruction from point cloud,” *arXiv preprint arXiv:1907.01142*, 2019.
- [284] E. Bae, X.-C. Tai, and W. Zhu, “Augmented Lagrangian method for an Euler’s elastica based segmentation model that promotes convex contours,” *Inverse Problems & Imaging*, vol. 11, no. 1, pp. 1–23, 2017.
- [285] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang, “Implicit, nonparametric shape reconstruction from unorganized points using a variational level set method,” *Computer Vision and Image Understanding*, vol. 80, no. 3, pp. 295–319, 2000.
- [286] R. Bracewell and R. Bracewell, *The Fourier Transform and Its Applications*, ser. Electrical Engineering Series. McGraw Hill, 2000, ISBN: 9780073039381.
- [287] X.-C. Tai, J. Hahn, and G. J. Chung, “A fast algorithm for Euler’s elastica model using augmented Lagrangian method,” *SIAM Journal on Imaging Sciences*, vol. 4, no. 1, pp. 313–344, 2011.

- [288] C. Y. Kao, S. Osher, and J. Qian, “Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations,” *Journal of Computational Physics*, vol. 196, no. 1, pp. 367–391, 2004.
- [289] J. Shen, S. H. Kang, and T. F. Chan, “Euler’s elastica and curvature-based inpainting,” *SIAM Journal on Applied Mathematics*, vol. 63, no. 2, pp. 564–592, 2003.
- [290] L.-J. Deng, R. Glowinski, and X.-C. Tai, “A new operator splitting method for the Euler elastica model for image smoothing,” *SIAM Journal on Imaging Sciences*, 2019.
- [291] W. Zhu, X.-C. Tai, and T. Chan, “Image segmentation using Euler’s elastica as the regularization,” *Journal of Scientific Computing*, vol. 57, no. 2, pp. 414–438, 2013.
- [292] W. Blaschke, “Über topologische fragen der differentialgeometrie.,” *Jahresbericht der Deutschen Mathematiker-Vereinigung*, vol. 38, pp. 193–205, 1929.
- [293] T. Willmore, “Mean curvature of immersed surfaces,” *An. Sti. Univ. Al. I. Cuza Iasi, Sec. I. a Mat.(NS)*, vol. 14, pp. 99–103, 1968.
- [294] W. Zhu and T. Chan, “Image denoising using mean curvature of image surface,” *SIAM Journal on Imaging Sciences*, vol. 5, no. 1, pp. 1–32, 2012.
- [295] E. Bae, X.-C. Tai, and Z. Wei, “Augmented lagrangian method for an euler’s elastica based segmentation model that promotes convex contours,” 2017.
- [296] M. Droske and A. Bertozzi, “Higher-order feature-preserving geometric regularization,” *SIAM Journal on Imaging Sciences*, vol. 3, no. 1, pp. 21–51, 2010.
- [297] Y. Gong and O. Goksel, “Weighted mean curvature,” *Signal Processing*, vol. 164, pp. 329–339, 2019.
- [298] X. Qiao, “The principle curvature-driven diffusion model for image de-noising,” in *3rd International Conference on Multimedia Technology (ICMT-13)*, Atlantis Press, 2013.
- [299] Y. Gong and I. F. Sbalzarini, “Local weighted gaussian curvature for image processing,” in *2013 IEEE International Conference on Image Processing*, IEEE, 2013, pp. 534–538.
- [300] B. Goldluecke and D. Cremers, “Introducing total curvature for image processing,” in *2011 International Conference on Computer Vision*, IEEE, 2011, pp. 1267–1274.
- [301] L. M. Lui, C. Wen, and X. Gu, “A conformal approach for surface inpainting,” *Inverse Problems and Imaging*, vol. 7, no. 3, pp. 863–884, 2013.

- [302] T. Schoenemann, S. Masnou, and D. Cremers, “The elastic ratio: Introducing curvature into ratio-based image segmentation,” *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2565–2581, 2011.
- [303] C. Brito-Loeza and K. Chen, “Multigrid method for a modified curvature driven diffusion model for image inpainting,” *Journal of Computational Mathematics*, pp. 856–875, 2008.
- [304] F. Yang, K. Chen, and B. Yu, “Homotopy method for a mean curvature-based denoising model,” *Applied Numerical Mathematics*, vol. 62, no. 3, pp. 185–200, 2012.
- [305] K. Bredies, T. Pock, and B. Wirth, “A convex, lower semicontinuous approximation of Euler’s elastica energy,” *SIAM Journal on Mathematical Analysis*, vol. 47, no. 1, pp. 566–613, 2015.
- [306] T. Schoenemann, F. Kahl, S. Masnou, and D. Cremers, “A linear framework for region-based image segmentation and inpainting involving curvature penalization,” *International Journal of Computer Vision*, vol. 99, no. 1, pp. 53–68, 2012.
- [307] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM, 1989, vol. 9.
- [308] H. F. Trotter, “On the product of semi-groups of operators,” *Proceedings of the American Mathematical Society*, vol. 10, no. 4, pp. 545–551, 1959.
- [309] X.-C. Tai and C. Wu, “Augmented Lagrangian method, dual methods and split Bregman iteration for ROF model,” in *International Conference on Scale Space and Variational Methods in Computer Vision*, Springer, 2009, pp. 502–513.
- [310] M. Yashtini and S. H. Kang, “A fast relaxed normal two split method and an effective weighted TV approach for Euler’s elastica image inpainting,” *SIAM Journal on Imaging Sciences*, vol. 9, no. 4, pp. 1552–1581, 2016.
- [311] Y. He, S. H. Kang, and H. Liu, “Curvature regularized surface reconstruction from point clouds,” *SIAM Journal on Imaging Sciences*, vol. 13, no. 4, pp. 1834–1859, 2020.
- [312] S. Osher, R. Fedkiw, and K. Piechor, “Level set methods and dynamic implicit surfaces,” *Appl. Mech. Rev.*, vol. 57, no. 3, B15–B15, 2004.
- [313] I. M. Mladenov and J. Oprea, “The mylar ballon: New viewpoints and generalizations,” in *Proceedings of the Eighth International Conference on Geometry, Integrability and Quantization*, Institute of Biophysics and Biomedical Engineering, Bulgarian Academy of Sciences, 2007, pp. 246–263.

- [314] R. Glowinski, S. J. Osher, and W. Yin, *Splitting Methods in Communication, Imaging, Science, and Engineering*. Springer, 2017.
- [315] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang, “A PDE-based fast local level set method,” *Journal of computational physics*, vol. 155, no. 2, pp. 410–438, 1999.
- [316] K. Zhang and H. Huang, “Underwater image transmission and blurred image restoration,” *Optical Engineering*, vol. 40, no. 6, 2001.
- [317] M. Yang and A. Sowmya, “An underwater color image quality evaluation metric,” *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 6062–6071, 2015.
- [318] H. Wen, Y. Tian, T. Huang, and W. Gao, “Single underwater image enhancement with a new optical model,” in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, IEEE, 2013, pp. 753–756.
- [319] Y.-T. Peng and P. C. Cosman, “Underwater image restoration based on image blurriness and light absorption,” *IEEE transactions on image processing*, vol. 26, no. 4, pp. 1579–1594, 2017.
- [320] D. Berman, T. Treibitz, and S. Avidan, “Diving into haze-lines: Color restoration of underwater images,” in *Proc. British Machine Vision Conference (BMVC)*, vol. 1, 2017.
- [321] C. Li, C. Guo, W. Ren, R. Cong, J. Hou, S. Kwong, and D. Tao, “An underwater image enhancement benchmark dataset and beyond,” *IEEE Transactions on Image Processing*, vol. 29, pp. 4376–4389, 2019.
- [322] H. Koschmieder, “Theorie der horizontalen sichtweite,” *Beitrage zur Physik der freien Atmosphere*, pp. 33–53, 1924.
- [323] D. Swinehart, “The Beer-Lambert law,” *Journal of chemical education*, vol. 39, no. 7, p. 333, 1962.
- [324] N. Jerlov, “Irradiance optical classification,” *Optical Oceanography*, pp. 118–120, 1968.
- [325] J. Beck, “The perception of surface color,” *Scientific American*, vol. 233, no. 2, pp. 62–77, 1975.
- [326] G. Finlayson and S. Hordley, “Improving gamut mapping color constancy,” *IEEE Transactions on Image Processing*, vol. 9, no. 10, pp. 1774–1783, 2000.

- [327] A. Moore, J. Allman, and R. M. Goodman, “A real-time neural system for color constancy,” *IEEE Transactions on Neural networks*, vol. 2, no. 2, pp. 237–247, 1991.
- [328] G. D. Finlayson, M. S. Drew, and B. V. Funt, “Spectral sharpening: Sensor transformations for improved color constancy,” *JOSA A*, vol. 11, no. 5, pp. 1553–1563, 1994.
- [329] Z.-u. Rahman, D. J. Jobson, and G. A. Woodell, “Multi-scale retinex for color image enhancement,” in *Proceedings of 3rd IEEE International Conference on Image Processing*, IEEE, vol. 3, 1996, pp. 1003–1006.
- [330] A. B. Petro, C. Sbert, and J.-M. Morel, “Multiscale Retinex,” *Image Processing On Line*, pp. 71–88, 2014.
- [331] E. Provenzi, M. Fierro, A. Rizzi, L. De Carli, D. Gadia, and D. Marini, “Random spray retinex: A new retinex implementation to investigate the local properties of the model,” *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 162–171, 2006.
- [332] D. Zosso, G. Tran, and S. J. Osher, “Non-local retinex—a unifying framework and beyond,” *SIAM Journal on Imaging Sciences*, vol. 8, no. 2, pp. 787–826, 2015.
- [333] J. M. Morel, A. B. Petro, and C. Sbert, “A PDE formalization of retinex theory,” *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2825–2837, 2010.
- [334] J. J. Gibson, “Adaptation with negative after-effect,” *Psychological review*, vol. 44, no. 3, p. 222, 1937.
- [335] S. C. Belmore and S. K. Shevell, “Very-long-term and short-term chromatic adaptation: Are their influences cumulative?” *Vision research*, vol. 51, no. 3, pp. 362–366, 2011.
- [336] K. E. Tregillus and S. A. Engel, “Long-term adaptation to color,” *Current Opinion in Behavioral Sciences*, vol. 30, pp. 116–121, 2019.
- [337] A. N. Tikhonov, “On the solution of ill-posed problems and the method of regularization,” in *Doklady Akademii Nauk*, Russian Academy of Sciences, vol. 151, 1963, pp. 501–504.
- [338] J. McCann, “Local/global mechanisms for color constancy,” *Die Farbe*, vol. 34, pp. 275–283, 1987.
- [339] T. Azetsu and N. Suetake, “Hue-preserving image enhancement in CIELAB color space considering color gamut,” *Optical Review*, vol. 26, no. 2, pp. 283–294, 2019.

- [340] C. Ueda, T. Azetsu, N. Suetake, and E. Uchino, "Lightness and chroma enhancement for food images considering helmholtz–kohlrausch effect," *Optical Review*, vol. 24, no. 3, pp. 301–309, 2017.
- [341] S. K. Shevell, "The time course of chromatic adaptation," *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, vol. 26, no. S1, S170–S173, 2001.
- [342] G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin institute*, vol. 310, no. 1, pp. 1–26, 1980.
- [343] H. R. Wilson and J. D. Cowan, "Excitatory and inhibitory interactions in localized populations of model neurons," *Biophysical journal*, vol. 12, no. 1, pp. 1–24, 1972.
- [344] M. Bertalmío, "From image processing to computational neuroscience: A neural model based on histogram equalization," *Frontiers in computational neuroscience*, vol. 8, p. 71, 2014.
- [345] K. McLaren, "CIELAB hue-angle anomalies at low tristimulus ratios," *Color Research & Application*, vol. 5, no. 3, pp. 139–143, 1980.
- [346] F. Ebner and M. D. Fairchild, "Finding constant hue surfaces in color space," in *Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts III*, International Society for Optics and Photonics, vol. 3300, 1998, pp. 107–117.
- [347] G. J. Braun, M. D. Fairchild, and F. Ebner, "Color gamut mapping in a hue-linearized CIELAB color space," in *Color and imaging conference*, Society for Imaging Science and Technology, vol. 1998, 1998, pp. 163–168.
- [348] N. Moroney, "A hypothesis regarding the poor blue constancy of CIELAB," *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, vol. 28, no. 5, pp. 371–378, 2003.
- [349] M. R. Luo, G. Cui, and B. Rigg, "The development of the CIE 2000 colour-difference formula: CIEDE2000," *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, vol. 26, no. 5, pp. 340–350, 2001.

- [350] R. L. Donofrio, "The Helmholtz-Kohlrausch effect," *Journal of the Society for Information Display*, vol. 19, no. 10, pp. 658–664, 2011.
- [351] M. D. Fairchild and E. Pirrotta, "Predicting the lightness of chromatic object colors using CIELAB," *Color Research & Application*, vol. 16, no. 6, pp. 385–393, 1991.
- [352] X. Zhao, T. Jin, and S. Qu, "Deriving inherent optical properties from background color and underwater image enhancement," *Ocean Engineering*, vol. 94, no. jan.15, pp. 163–172,
- [353] N. Limare, J.-L. Lisani, J.-M. Morel, A. B. Petro, and C. Sbert, "Simplest Color Balance," *Image Processing On Line*, vol. 1, pp. 297–315, 2011.
- [354] P. Getreuer, "Automatic Color Enhancement (ACE) and its Fast Implementation," *Image Processing On Line*, vol. 2, pp. 266–277, 2012.
- [355] J. G. Gomila Salas and J. L. Lisani, "Local Color Correction," *Image Processing On Line*, vol. 1, pp. 260–280, 2011.
- [356] K. Panetta, C. Gao, and S. Agaian, "Human-visual-system-inspired underwater image quality measures," *IEEE Journal of Oceanic Engineering*, vol. 41, no. 3, pp. 541–551, 2016.
- [357] H. G. Bock, "Recent advances in parameter identification techniques for ODE," in *Numerical treatment of inverse problems in differential and integral equations*, Springer, 1983, pp. 95–121.
- [358] T. Müller and J. Timmer, "Parameter identification techniques for partial differential equations," *International Journal of Bifurcation and Chaos*, vol. 14, no. 06, pp. 2053–2060, 2004.
- [359] E. Baake, M. Baake, H. Bock, and K. Briggs, "Fitting ordinary differential equations to chaotic data," *Physical Review A*, vol. 45, no. 8, p. 5524, 1992.
- [360] T. G. Müller and J. Timmer, "Fitting parameters in partial differential equations from partially observed noisy data," *Physica D: Nonlinear Phenomena*, vol. 171, no. 1-2, pp. 1–7, 2002.
- [361] H. G. Bock, "Numerical treatment of inverse problems in chemical reaction kinetics," in *Modelling of chemical reaction systems*, Springer, 1981, pp. 102–125.
- [362] U. Parlitz and C. Merkwirth, "Prediction of spatiotemporal time series based on reconstructed local states," *Physical review letters*, vol. 84, no. 9, p. 1890, 2000.

- [363] M. Bär, R. Hegger, and H. Kantz, “Fitting partial differential equations to space-time dynamics,” *Physical Review E*, vol. 59, no. 1, p. 337, 1999.
- [364] J. Bongard and H. Lipson, “Automated reverse engineering of nonlinear dynamical systems,” *Proceedings of the National Academy of Sciences*, vol. 104, no. 24, pp. 9943–9948, 2007.
- [365] M. Schmidt and H. Lipson, “Distilling free-form natural laws from experimental data,” *science*, vol. 324, no. 5923, pp. 81–85, 2009.
- [366] S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Discovering governing equations from data by sparse identification of nonlinear dynamical systems,” *Proceedings of the national academy of sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.
- [367] H. Schaeffer, “Learning partial differential equations via data discovery and sparse optimization,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2197, p. 20 160 446, 2017.
- [368] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Data-driven discovery of partial differential equations,” *Science Advances*, vol. 3, no. 4, e1602614, 2017.
- [369] S. H. Kang, W. Liao, and Y. Liu, “Ident: Identifying differential equations with numerical time evolution,” *arXiv preprint arXiv:1904.03538*, 2019.
- [370] E. Kaiser, J. N. Kutz, and S. L. Brunton, “Sparse identification of nonlinear dynamics for model predictive control in the low-data limit,” *Proceedings of the Royal Society A*, vol. 474, no. 2219, p. 20 180 335, 2018.
- [371] J.-C. Loiseau and S. L. Brunton, “Constrained sparse galerkin regression,” *Journal of Fluid Mechanics*, vol. 838, pp. 42–67, 2018.
- [372] N. M. Mangan, J. N. Kutz, S. L. Brunton, and J. L. Proctor, “Model selection for dynamical systems via sparse regression and information criteria,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2204, p. 20 170 009, 2017.
- [373] G. Tran and R. Ward, “Exact recovery of chaotic systems from highly corrupted data,” *Multiscale Modeling & Simulation*, vol. 15, no. 3, pp. 1108–1129, 2017.
- [374] H. Schaeffer, G. Tran, and R. Ward, “Extracting sparse high-dimensional dynamics from limited data,” *SIAM Journal on Applied Mathematics*, vol. 78, no. 6, pp. 3279–3295, 2018.
- [375] M. M. Zhang, H. Lam, and L. Lin, “Robust and parallel bayesian model selection,” *Computational Statistics & Data Analysis*, vol. 127, pp. 229–247, 2018.

- [376] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher, “Sparse dynamics for partial differential equations,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 17, pp. 6634–6639, 2013.
- [377] M. Bongini, M. Fornasier, M. Hansen, and M. Maggioni, “Inferring interaction rules from observations of evolutive systems i: The variational approach,” *Mathematical Models and Methods in Applied Sciences*, vol. 27, no. 05, pp. 909–951, 2017.
- [378] F. Lu, M. Zhong, S. Tang, and M. Maggioni, “Nonparametric inference of interaction laws in systems of agents from trajectory data,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 29, pp. 14 424–14 433, 2019.
- [379] Z. Long, Y. Lu, X. Ma, and B. Dong, “PDE-net: Learning PDEs from data,” *arXiv preprint arXiv:1710.09668*, 2017.
- [380] Z. Long, Y. Lu, and B. Dong, “Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network,” *Journal of Computational Physics*, vol. 399, p. 108 925, 2019.
- [381] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations,” *arXiv preprint arXiv:1711.10561*, 2017.
- [382] T. Qin, K. Wu, and D. Xiu, “Data driven governing equations approximation using deep neural networks,” *Journal of Computational Physics*, 2019.
- [383] M. Raissi and G. E. Karniadakis, “Hidden physics models: Machine learning of nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 357, pp. 125–141, 2018.
- [384] Y. Khoo and L. Ying, “SwitchNet: A neural network model for forward and inverse scattering problems,” *arXiv preprint arXiv:1810.09675*, 2018.
- [385] B. Lusch, J. N. Kutz, and S. L. Brunton, “Deep learning for universal linear embeddings of nonlinear dynamics,” *Nature communications*, vol. 9, no. 1, p. 4950, 2018.
- [386] Y. He, S. H. Kang, W. Liao, H. Liu, and Y. Liu, “Robust PDE identification from noisy data,” *arXiv preprint arXiv:2006.06557*, 2020.
- [387] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, “Uniformly high order accurate essentially non-oscillatory schemes, iii,” in *Upwind and high-resolution schemes*, Springer, 1987, pp. 218–290.

- [388] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Pub. San Diego, 1997.
- [389] P. Lancaster and K. Salkauskas, "Surfaces generated by moving least squares methods," *Mathematics of computation*, vol. 37, no. 155, pp. 141–158, 1981.
- [390] H. Wendland, "Local polynomial reproduction and moving least squares approximation," *IMA Journal of Numerical Analysis*, vol. 21, no. 1, pp. 285–300, 2001.
- [391] E. J. Candés, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.
- [392] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [393] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [394] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.
- [395] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
- [396] M. Tham, "Dealing with measurement noise. moving average filter," *Chemical Engineering and Advanced Materials, University of Newcastle upon Tyne*, 1998.
- [397] P. Craven and G. Wahba, "Smoothing noisy data with spline functions," *Numerische mathematik*, vol. 31, no. 4, pp. 377–403, 1978.
- [398] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.
- [399] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE transactions on information theory*, vol. 47, no. 7, pp. 2845–2862, 2001.
- [400] D. L. Donoho, M. Elad, and V. N. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions on information theory*, vol. 52, no. 1, pp. 6–18, 2005.

- [401] A. Feuer and A. Nemirovski, “On sparse representation in pairs of bases,” *IEEE Transactions on Information Theory*, vol. 49, no. 6, pp. 1579–1581, 2003.
- [402] E. J. Candes and T. Tao, “Decoding by linear programming,” *IEEE transactions on information theory*, vol. 51, no. 12, pp. 4203–4215, 2005.
- [403] K. Knight and W. Fu, “Asymptotics for lasso-type estimators,” *Annals of statistics*, pp. 1356–1378, 2000.
- [404] J. A. Tropp, “Just relax: Convex programming methods for identifying sparse signals in noise,” *IEEE transactions on information theory*, vol. 52, no. 3, pp. 1030–1051, 2006.
- [405] P. Zhao and B. Yu, “On model selection consistency of lasso,” *Journal of Machine learning research*, vol. 7, no. Nov, pp. 2541–2563, 2006.
- [406] J.-J. Fuchs, “Recovery of exact sparse representations in the presence of bounded noise,” *IEEE Transactions on Information Theory*, vol. 51, no. 10, pp. 3601–3608, 2005.
- [407] M. J. Wainwright, “Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1 -constrained quadratic programming (Lasso),” *IEEE transactions on information theory*, vol. 55, no. 5, pp. 2183–2202, 2009.
- [408] J. Jia, K. Rohe, and B. Yu, “The lasso under poisson-like heteroscedasticity,” *Statistica Sinica*, pp. 99–118, 2013.
- [409] N. Meinshausen, P. Bühlmann, *et al.*, “High-dimensional graphs and variable selection with the lasso,” *The annals of statistics*, vol. 34, no. 3, pp. 1436–1462, 2006.
- [410] P. Ravikumar, G. Raskutti, M. J. Wainwright, and B. Yu, “Model selection in gaussian graphical models: High-dimensional consistency of ℓ_1 -regularized mle,” in *NIPS*, 2008, pp. 1329–1336.
- [411] P. Ravikumar, M. J. Wainwright, and J. D. Lafferty, “High-dimensional ising model selection using ℓ_1 -regularized logistic regression,” *The Annals of Statistics*, vol. 38, no. 3, pp. 1287–1319, 2010.
- [412] J. Fan and J. Lv, “A selective overview of variable selection in high dimensional feature space,” *Statistica Sinica*, vol. 20, no. 1, p. 101, 2010.
- [413] J. Fan, T. Gasser, I. Gijbels, M. Brockmann, and J. Engel, “Local polynomial regression: Optimal kernels and asymptotic minimax efficiency,” *Annals of the Institute of Statistical Mathematics*, vol. 49, no. 1, pp. 79–99, 1997.

- [414] J. Fan, *Local polynomial modelling and its applications: monographs on statistics and applied probability* 66. Routledge, 2018.
- [415] Y.-p. Mack and B. W. Silverman, “Weak and strong uniform consistency of kernel regression estimates,” *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 61, no. 3, pp. 405–415, 1982.
- [416] G. Tusnády, “A remark on the approximation of the sample df in the multidimensional case,” *Periodica Mathematica Hungarica*, vol. 8, no. 1, pp. 53–55, 1977.
- [417] H. Liang and H. Wu, “Parameter estimation for differential equation models using a framework of measurement error in regression models,” *Journal of the American Statistical Association*, vol. 103, no. 484, pp. 1570–1583, 2008.
- [418] E. Masry, “Multivariate local polynomial regression for time series: Uniform strong consistency and rates,” *Journal of Time Series Analysis*, vol. 17, no. 6, pp. 571–599, 1996.
- [419] Y. Li, T. Hsing, *et al.*, “Uniform convergence rates for nonparametric regression and principal component analysis in functional/longitudinal data,” *The Annals of Statistics*, vol. 38, no. 6, pp. 3321–3351, 2010.
- [420] B. W. Silverman, “Weak and strong uniform consistency of the kernel estimate of a density and its derivatives,” *The Annals of Statistics*, pp. 177–184, 1978.
- [421] T. Tao, *Topics in random matrix theory*. American Mathematical Soc., 2012, vol. 132.
- [422] K. R. Davidson and S. J. Szarek, “Local operator theory, random matrices and banach spaces,” *Handbook of the geometry of Banach spaces*, vol. 1, no. 317–366, p. 131, 2001.
- [423] D. L. Bailey, M. N. Maisey, D. W. Townsend, and P. E. Valk, *Positron emission tomography*. Springer, 2005, vol. 2.
- [424] C. Liu, L. A. Pierce II, A. M. Alessio, and P. E. Kinahan, “The impact of respiratory motion on tumor quantification and delineation in static PET/CT imaging,” *Physics in Medicine & Biology*, vol. 54, no. 24, p. 7345, 2009.
- [425] A. Rahmim, M. A. Lodge, N. A. Karakatsanis, V. Y. Panin, Y. Zhou, A. McMillan, S. Cho, H. Zaidi, M. E. Casey, and R. L. Wahl, “Dynamic whole-body PET imaging: Principles, potentials and applications,” *European journal of nuclear medicine and molecular imaging*, vol. 46, no. 2, pp. 501–518, 2019.
- [426] O. Muzik, T. J. Mangner, W. R. Leonard, A. Kumar, J. Janisse, and J. G. Granneman, “ ^{15}O PET measurement of blood flow and oxygen consumption in cold-

- activated human brown fat,” *Journal of Nuclear Medicine*, vol. 54, no. 4, pp. 523–531, 2013.
- [427] B. M. Larimer, E. Wehrenberg-Klee, A. Caraballo, and U. Mahmood, “Quantitative CD3 PET imaging predicts tumor growth response to anti-CTLA-4 therapy,” *Journal of Nuclear Medicine*, vol. 57, no. 10, pp. 1607–1611, 2016.
 - [428] M. Hatt, D. Visvikis, O. Pradier, and C. Cheze-Le Rest, “Baseline 18 F-FDG PET image-derived parameters for therapy response prediction in oesophageal cancer,” *European journal of nuclear medicine and molecular imaging*, vol. 38, no. 9, pp. 1595–1606, 2011.
 - [429] E. Inglese, L. Leva, R. Matheoud, G. Sacchetti, C. Secco, P. Gandolfo, M. Brambilla, and G. Sambuceti, “Spatial and temporal heterogeneity of regional myocardial uptake in patients without heart disease under fasting conditions on repeated whole-body 18F-FDG PET/CT,” *Journal of Nuclear Medicine*, vol. 48, no. 10, pp. 1662–1669, 2007.
 - [430] P. Videbech, “PET measurements of brain glucose metabolism and blood flow in major depressive disorder: A critical review,” *Acta Psychiatrica Scandinavica*, vol. 101, no. 1, pp. 11–20, 2000.
 - [431] S. A. Nehmeh, Y. E. Erdi, K. E. Rosenzweig, H. Schoder, S. M. Larson, O. D. Squire, and J. L. Humm, “Reduction of respiratory motion artifacts in PET imaging of lung cancer by respiratory correlated dynamic PET: Methodology and comparison with respiratory gated PET,” *Journal of Nuclear Medicine*, vol. 44, no. 10, pp. 1644–1648, 2003.
 - [432] M. Lortie, R. S. Beanlands, K. Yoshinaga, R. Klein, J. N. DaSilva, and R. A. DeKemp, “Quantification of myocardial blood flow with 82 Rb dynamic PET imaging,” *European journal of nuclear medicine and molecular imaging*, vol. 34, no. 11, pp. 1765–1774, 2007.
 - [433] M. D. Normandin, W. K. Schiffer, and E. D. Morris, “A linear model for estimation of neurotransmitter response profiles from dynamic PET data,” *Neuroimage*, vol. 59, no. 3, pp. 2689–2699, 2012.
 - [434] M. E. Kamasak, C. A. Bouman, E. D. Morris, and K. Sauer, “Direct reconstruction of kinetic parameter images from dynamic PET data,” *IEEE transactions on medical imaging*, vol. 24, no. 5, pp. 636–650, 2005.
 - [435] K.-P. Wong, D. Feng, S. R. Meikle, and M. J. Fulham, “Segmentation of dynamic PET images using cluster analysis,” *IEEE Transactions on nuclear science*, vol. 49, no. 1, pp. 200–207, 2002.

- [436] F. Hashimoto, H. Ohba, K. Ote, A. Teramoto, and H. Tsukada, “Dynamic PET image denoising using deep convolutional neural networks without prior training datasets,” *IEEE Access*, vol. 7, pp. 96 594–96 603, 2019.
- [437] J. Cui, H. Yu, S. Chen, Y. Chen, and H. Liu, “Simultaneous estimation and segmentation from projection data in dynamic PET,” *Medical physics*, vol. 46, no. 3, pp. 1245–1259, 2019.
- [438] J. Cui, X. Liu, Y. Wang, and H. Liu, “Deep reconstruction model for dynamic PET images,” *PloS one*, vol. 12, no. 9, e0184667, 2017.
- [439] L. Lu, N. A. Karakatsanis, J. Tang, W. Chen, and A. Rahmim, “3.5 D dynamic PET image reconstruction incorporating kinetics-based clusters,” *Physics in Medicine & Biology*, vol. 57, no. 15, p. 5035, 2012.
- [440] M. N. Wernick, E. J. Infusino, and M. Milosevic, “Fast spatio-temporal image reconstruction for dynamic PET,” *IEEE transactions on medical imaging*, vol. 18, no. 3, pp. 185–195, 1999.
- [441] M. Burger, C. Rossmanith, and X. Zhang, “Simultaneous reconstruction and segmentation for dynamic SPECT imaging,” *Inverse Problems*, vol. 32, no. 10, p. 104 002, 2016.
- [442] Q. Ding, M. Burger, and X. Zhang, “Dynamic SPECT reconstruction with temporal edge correlation,” *Inverse Problems*, vol. 34, no. 1, p. 014 005, 2017.
- [443] Z. Zhang and H. Liu, “Nonlocal total variation based dynamic PET image reconstruction with low-rank constraints,” *Physica Scripta*, vol. 94, no. 6, p. 065 202, 2019.
- [444] T. Yokota, K. Kawai, M. Sakata, Y. Kimura, and H. Hontani, “Dynamic PET image reconstruction using nonnegative matrix factorization incorporated with deep image prior,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3126–3135.
- [445] B. Wang and H. Liu, “FBP-Net for direct reconstruction of dynamic PET images,” *Physics in Medicine & Biology*, vol. 65, no. 23, p. 235 008, 2020.
- [446] Y. He, S. H. Kang, Q. Ding, and X. Zhang, “Deep dynamic pet reconstruction by spatial and temporal information synthesis,” *Submitted*, 2021.
- [447] R. E. Carson, “Tracer kinetic modeling in PET,” in *Positron Emission Tomography*, Springer, 2005, pp. 127–159.

- [448] S. S. Dragomir, “Reverses of the Schwarz inequality in inner product spaces and applications,” *Research report collection*, vol. 7, no. 1, 2004.
- [449] Y. Nomura, Y. Asano, J. Shinoda, H. Yano, Y. Ikegame, T. Kawasaki, N. Nakayama, T. Maruyama, Y. Muragaki, and T. Iwama, “Characteristics of time-activity curves obtained from dynamic ^{11}C -methionine PET in common primary brain tumors,” *Journal of neuro-oncology*, vol. 138, no. 3, pp. 649–658, 2018.
- [450] Q. Ding, Y. Zan, Q. Huang, and X. Zhang, “Dynamic spect reconstruction from few projections: A sparsity enforced matrix factorization approach,” *Inverse Problems*, vol. 31, no. 2, p. 025 004, 2015.
- [451] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising,” *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.
- [452] H. Chen, Y. Zhang, Y. Chen, J. Zhang, W. Zhang, H. Sun, Y. Lv, P. Liao, J. Zhou, and G. Wang, “Learn: Learned experts’ assessment-based reconstruction network for sparse-data ct,” *IEEE transactions on medical imaging*, vol. 37, no. 6, pp. 1333–1347, 2018.
- [453] B. Johansson, T. Elfving, V. Kozlov, Y. Censor, P.-E. Forssén, and G. Granlund, “The application of an oblique-projected landweber method to a model of supervised learning,” *Mathematical and computer modelling*, vol. 43, no. 7-8, pp. 892–909, 2006.
- [454] Z. Kotevski and P. Mitrevski, “Experimental comparison of PSNR and SSIM metrics for video quality estimation,” in *International Conference on ICT Innovations*, Springer, 2009, pp. 357–366.
- [455] A. Dimitrakopoulou-Strauss, L. Pan, and C. Sachpekidis, “Kinetic modeling and parametric imaging with dynamic pet for oncological applications: General considerations, current clinical applications, and future perspectives,” *European Journal of Nuclear Medicine and Molecular Imaging*, pp. 1–19, 2020.
- [456] L. Latecki, R. Lakamper, and T. Eckhardt, “Shape descriptors for non-rigid shapes with a single closed contour,” in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, IEEE, vol. 1, 2000, pp. 424–429.
- [457] P.-Å. Wedin, “Perturbation theory for pseudo-inverses,” *BIT Numerical Mathematics*, vol. 13, no. 2, pp. 217–232, 1973.
- [458] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

- [459] M. Rosenblatt, “Remarks on a multivariate transformation,” *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 470–472, 1952.
- [460] A. Winkelbauer, “Moments and absolute moments of the normal distribution,” *arXiv preprint arXiv:1209.4340*, 2012.
- [461] J. Bretagnolle and P. Massart, “Hungarian constructions from the nonasymptotic viewpoint,” *The Annals of Probability*, pp. 239–256, 1989.

VITA

Yuchen He was born on October 6, 1990, in Chongqing, China. He got his Bachelor degree in Mathematical Statistics from Chongqing University, Chongqing, China in 2009. and Master degree in Statistics from Columbia University, New York, New York, USA in 2014. He is pursuing his Ph.D. in Mathematics at Georgia Institute of Technology, Atlanta, Georgia, USA starting from 2016, and he is anticipated to graduate in May, 2021.

Yuchen He likes painting during his leisure time. In fact, he was trained as an art student since he was 4 years old, and he decided to pursue the beauty of mathematics since he entered the college. In his early stage of Ph.D. life, Yuchen He was interested in algebraic geometry and explored this area under the guidance of Professor Kirsten Wickelgren. After learning more about numerical mathematics and their applications in various interesting real-life problems, he decided to focus on applied mathematics under the supervision of Professor Sung Ha Kang. Since then, he gained experience of working on problems with diverse applications using various mathematical techniques.

During his Ph.D. journey, he has been active in attending many academic events. He attended the 2017 Joint Mathematics Meetings in Atlanta, GA, USA. He participated the CIRM pre-school for a thematic trimester in Institut Henri Poincaré, Centre International de Recontres Mathématiques, Luminy, Marseille, France in 2019. From Jun 2019 to Jul 2019, he went to Germany for the Seventh International Conference on Scale Space and Variational Methods in Computer Vision and presented a poster. In Jan 2020, supported by the Chateaubriand Fellowship, he went to École normale supérieure Paris-Saclay, Cachan, Paris, France, working with Professor Jean-Michel Morel. He also gave a talk at Research Horizon Seminar at Georgia Institute of Technology in Nov. 2020.

Besides mathematics and painting, he also likes learning foreign languages, swimming, listening to music.